October 1980
USA $2.95/DM 9.80

kilobaud

# MICROCOMPUTING

## Will Fate Shortchange Your Family's Future?

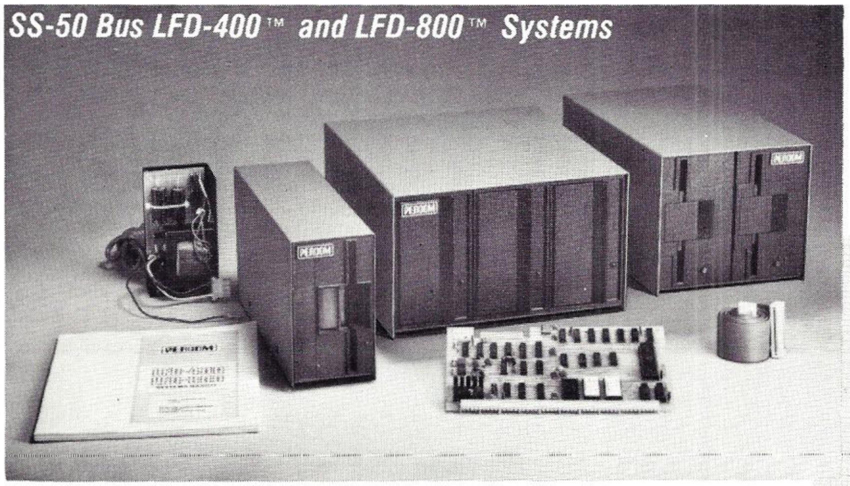
Computerize Your Estate Planning. Pg. 31


Picture-perfect SWTP program □ Z-80 upgrade for the H8 □ PET Mini Monitor unveiled □ Reining in Apple video □ Subscription-handling with the OSI □ Calling your computer by phone □ CP/M hard-copy secrets unlocked □ A pep pill for Microsoft BASIC □ New NSC800 . . . and more.

10
0 74470 65945

# A Few Extraordinary Products for Your 6800/6809 Computer

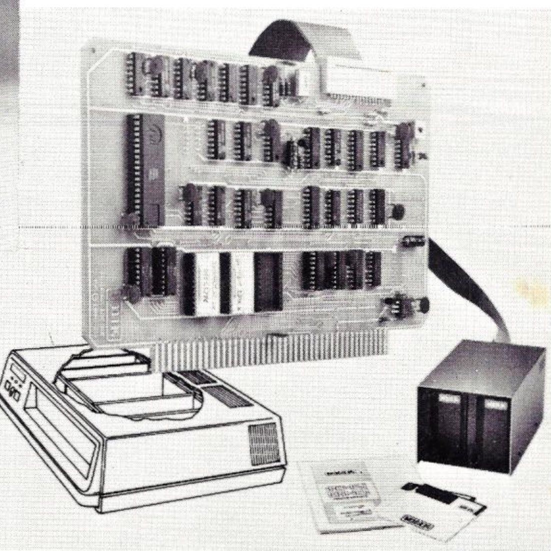
## From Percom . . .



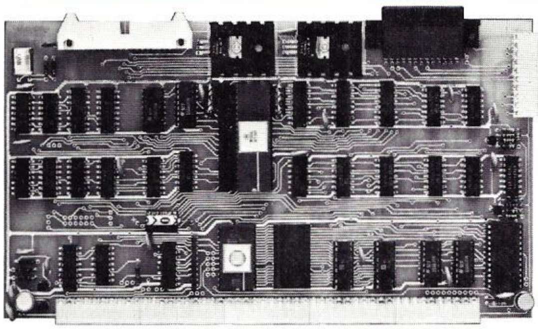

### Low Cost Mini-Disk Storage in the Size You Want ✓14

Percom mini-disk systems start as low as $599.95, ready to plug in and run. You can't get better quality or a broader selection of disk software from any other microcomputer disk system manufacturer — at any price!

**Features**: 1-, 2- and 3-drive systems in 40- and 77-track versions store 102K- to 591K-bytes of random access data on-line • controllers include explicit clock/data separation circuit, motor inactivity time-out circuit, buffered control lines and other mature design concepts • ROM DOS included with SS-50 bus version — optional DOSs for EXORciser* bus • extra PROM sockets on-board • EXORciser* bus version has 1K-byte RAM • supported by extended disk operating systems; assemblers and other program development/debugging aids; BASIC, FORTRAN, Pascal and SPL/M languages; and, business application programs.



*EXORciser* Bus LFD-400EX™ -800EX™ Systems*



### The SBC/9™. A "10" By Any Measure. ✓13

The Percom SBC/9™ is an SS-50 bus compatible, stand-alone Single-Board Computer. Configured for the 6809 microprocessor, the SBC/9™ also accommodates a 6802 without any modification. You can have state-of-the-art capability of the '09. Or put to work the enormous selection of 6800-coded programs that run on the '02.

The SBC/9™ includes PSYMON™, an easily extended 1-Kbyte ROM OS. Other features include:

- Total compatibility with the SS-50 bus. Requires no changes to the motherboard, memory or I/O.
- Serial port includes bit-rate generator. RS-232-C compatible with optional subminiature 'D' connector installed. 10-pin Molex connector provided.
- Eight-bit, non-latched, bidirectional parallel port is multi-address extension of system bus. Spans a 30-address field; accommodates an exceptional variety of peripheral devices. Connector is optional.
- Includes 1-Kbyte of static RAM.
- Costs only $199.95 with PSYMON™ and comprehensive users manual that includes source listing of PSYMON™.

### Versatile Mother Board, Full-Feature Prototyping Boards ✓15

Printed wiring is easily soldered tin-lead plating. Substrates are glass-epoxy. Prototyping cards provide for power regulators and distributed capacitor bypassing, accommodate 14-, 16-, 24- and 40-pin DIP sockets. Prototyping boards include bus connectors, other connectors and sockets are optional.

**MOTHERBOARD** — accommodates five SS-50 bus cards, and may itself be plugged into an SS-50 bus. Features wide-trace conductors. Price: $21.95

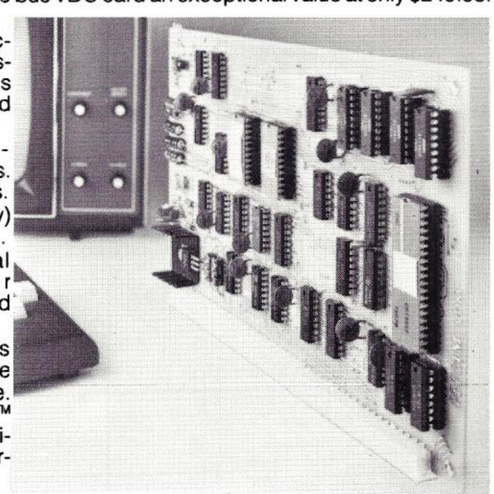
**SS-50 BUS CARD** — accommodates 34- and 50-pin ribbon connectors on top edge, 10-pin Molex connector on side edge. Price: $24.95.

**SS-30 BUS CARD** — 1¼-inch higher than SWTP I/O card, accommodates 34-pin ribbon connector and 12-pin Molex connector on top edge. Price: $14.95.

### The Electric Window™: Instant, Real-Time Video Display Control ✓16

Memory residency and outstanding software control of display format and characters make this SS-50 bus VDC card an exceptional value at only $249.95. Other features:

- Generates 128 characters including all ASCII displayable characters plus selected Greek letters and other special symbols.
- Well-formed, easy-to-read 7x12-dot characters. True baseline descenders.
- Character-store (display) memory included on card.
- Provision for optional character generator EPROM for user defined symbols.
- Comprehensive users manual includes source listing of Driver software. Driver — called WINDEX™ — is also available on minidiskette through the Percom Users Group.



™ trademark of Percom Data Company, Inc.
* trademark of the Motorola Corporation.

**Prices and specifications subject to change without notice.**

PERCOM

PERCOM DATA COMPANY, INC.
211 N. KIRBY GARLAND, TEXAS 75042
(214) 272-3421

**Products are available at Percom dealers nationwide. Call toll-free, 1-800-527-1592, for the address of your nearest dealer, or to order direct.**

# micro info

$ This symbol next to a title in the table of contents indicates that the article is a business-application article.

## Manuscripts

Contributions in the form of manuscripts with drawings and/or photographs are welcome and will be considered for possible publication. We can assume no responsibility for loss or damage to any material. Please enclose a self-addressed, stamped envelope with each submission. Payment for the use of any unsolicited material will be made upon acceptance. All contributions should be directed to the *Microcomputing* editorial offices. "How to Write for Microcomputing" guidelines are available upon request.


## Editorial Offices:

Pine Street
Peterborough NH 03458
Phone: 603-924-3873

## Advertising Offices:

Elm Street
Peterborough NH 03458
Phone: 603-924-7138

## Circulation Offices:

Elm Street
Peterborough NH 03458
Phone: 603-924-7296


## To subscribe, renew or change an address:

Write to *Microcomputing*, Subscription Department, PO Box 997, Farmingdale NY 11737. For renewals and changes of address, include the address label from your most recent issue of *Microcomputing*. For gift subscriptions, include your name and address as well as those of gift recipients. Postmaster: Send form #3579 to *Microcomputing*, Subscription Services, PO Box 997, Farmingdale NY 11737.

## Subscription problem or question:

Write to *Microcomputing*, Subscription Department, PO Box 997, Farmingdale NY 11737. Please include an address label.


Kilobaud *Microcomputing* (ISSN 0192-4575) is published monthly by Wayne Green, Inc., 80 Pine St., Peterborough NH 03458. Subscription rates in U.S. are $25 for one year and $53 for three years. In Canada: $27 for one year only, U.S. funds. Foreign subscriptions (surface mail)—$35 for one year only, U.S. funds. Foreign air mail subscriptions—$62 for one year only, U.S. funds. In Europe, contact: Monika Nedela, Markstr. 3, D-7778 Markdorf, W. Germany. South African Distributor: KB Microcomputing, PO Box 782815, Sandton, South Africa 2146. Australian Distributor: Electronic Concepts, Attention: Rudi Hoess, 55 Clarence Street, Sidney 2000, Australia. Second-class postage paid at Peterborough NH 03458 and at additional mailing offices. Phone: 603-924-3873.




kilobaud
# MICROCOMPUTING T.M.


## contents: October '80

## ARTICLES



## DEPARTMENTS



Cover photo by Reese Fowler.

Wayne Green

# PUBLISHER'S REMARKS

## Five Years Ago, Five Years Hence

The microcomputer industry is just five years old this year, yet it has already seen some spectacular changes—and more are to come. In 1975 Mits introduced the first microcomputer kit to attract widespread attention. Before that, several small firms had kits using the 8008 chip, but the interested numbered in the hundreds rather than the thousands. It took Mits to break things loose.

By the end of that first year, Mits had been joined by Sphere and Southwest Technical Products, both using 6800-based systems. And by this time I had already put out four issues of *Byte*, the first magazine for the industry.

During 1976 the field expanded, with systems from Imsai, Processor Tech, Polymorphics, Wavemate, The Digital Group, OSI, Intelligent Systems, M&R (Astral 2000), Apple, ECD, TDL, Veras Systems, etc.

Of the 1975 firms, only SWTP is still around, though I have seen few of their systems in stores or ads in a long time, and interest in software for their system seems to have disappeared. Apple has outperformed the 1976 group, with OSI continuing to grow. The rest have either disappeared or virtually disappeared.

In 1977 we had more successful starts, with Commodore, along with Heath, coming in early, followed by Radio Shack in mid-year. Commodore threw away their lead and hundreds of millions of dollars in sales by refusing to back up their system with advertising and an aggressive technical team. Heath tried to make a go of it with only their 50 company-owned stores. This, plus their refusal to go with the S-100 bus, in my estimation, cost them tens of millions of dollars in sales.

Radio Shack had a tough row to hoe at first. Their store managers not only knew nothing about computers, but most of them had little understanding of audio or CB, so they were afraid of this new invention and completely unequipped to cope with customers asking questions about ROMs and RAMs. They did have several things going for them—a superb instruction book, written by David Lien, and massive television and newspaper advertising. They also benefited from the almost total consumer invisibility of Commodore and Heath.

Approaching 1981, we're looking into our crystal ball to see what the microcomputer business will look like in five more years. It's almost impossible to look with any clarity into the middle of next year, much less five years down the pike. There are too many variables. If we assume that there are not going to be any more quantum developments, such as the 8080 chip, perhaps we can gain enough perspective to figure out where we can take advantage of what is happening for our own benefit.

The first order of business in predicting the future is to start with the market, presuming that the industry will pursue the market, rather than the reverse. The first computerists were hobbyists. They had to be, because making a computer from a kit, with pathetic instructions and with a finished product that often had not even worked in prototype, took considerable skill to have any success. The early manufacturers worked on the principle that hobbyists would build the kits and figure out how to make them work, thus performing the last ten percent of the engineering for them. It turned out that this system worked just fine, though it severely frustrated several thousand hobbyists in the process.

As the field matured, the early hobbyists either got fed up with the expense and bum equipment and dropped out, or became dealers or manufacturers in the business. Many of them are still around, taking advantage of the things they learned during the first two years of microcomputers.

With the advent of complete systems sales, the electronic-type hobbyist decreased in importance and was replaced by a new type of hobbyist, one interested in *using* the computer rather than building it. These new hobbyists quickly became deeply involved in developing utilities, writing programs, developing games and generally finding out the capabilities of their systems and expanding them.

There is still a good market for teenagers interested in learning about computers. I think this will continue, but I would expect it to be a relatively limited market as compared to business and educational applications. There may be a few hundred thousand kids with $1000 to spend on a hobby, but are there much more than that? We'll see.

The businessman sees the computer as a way to save money and to get more done in less time. But he is still wary of microcomputers. . . and he should be. I am still awaiting articles written by businessmen who have invested in microcomputer systems and found them to be of distinct advantage to them. Most of the letters I receive are to the contrary, expressing frustration over delivery problems, over service miseries and over the state of the software available. I'll be more convinced that the industry has an honest bargain for business when I start getting articles by the dozens lauding systems and programs they are using.

Once we have systems that can actually be used by business to save money and time, I think they will sell in prodigious quantities. Once we have systems that are clearly of value to schools, I think we'll be selling millions of microcomputers.

---

Where do we stand with suppliers? Radio Shack is out in front selling about three times as many systems as Apple, the next firm in line. That's my guess. I've visited the Radio Shack production facility and seen what they are doing. Apple has not offered to let me see theirs, so I don't know what they are doing for sure. But the sales figures that I have seen and the polls of our readers indicate about a three to one advantage for Radio Shack at present. The new Apple III may make the Apple of more interest for business applications, if they are somehow able to come up with some software support. In my discussions with Apple, I have not been convinced that they are serious about software—but then, neither is Radio Shack.

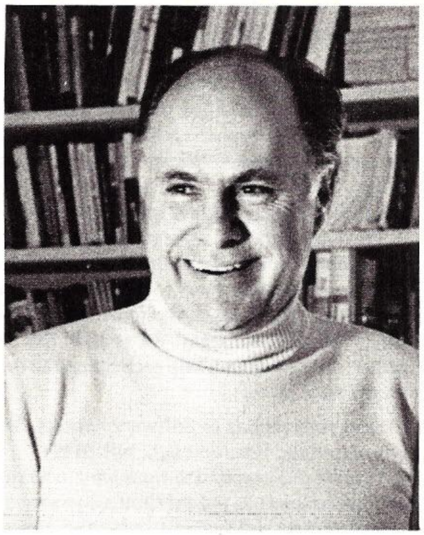
Commodore has new management and is starting to try to make up for lost time. They, too, have some new systems to try and keep up with the state of the art. It's quite a race, with Radio Shack fielding three new systems in July and one more promised for November, bringing their coverage up to six current models.

Other firms, such as Atari, Mattel and Bally, are all substantial firms, but none of them have shown any interest in more than minimal software for their systems, so I will be surprised if they do not waste a few million dollars before dropping out. They are mainly building extravagant toys to play games. I can't believe that this is a viable long-term market. I think that these firms are so involved with games that they got taken in by the term "personal computer" and think that there really is a market for personal toy computer systems. Computer games will always be popular on computers, but I don't think that many people (thousands maybe, but hundreds of thousands?) are going to spend $1000 to $2000 for a computer dedicated to games.

So we have several firms with good possibilities for growth over the next few years. But we also have a number of firms that seem to be heading in the wrong direction, and none that I have seen so far have any serious interest in bettering their sales potential with the needed software. I predict that one of the major firms will suddenly discover the importance of software and will quickly pass by all the others when they put this discovery to use. Until that happens, I see a battle taking place for a relatively small market for microcomputers. I don't think the market will increase until there is a large number of programs to support the systems.

## The Detroit Syndrome

The Japanese are getting ready to pounce, and I suspect that their marketing plan is going



to be a lot more clever than those we have seen from many of the American firms. The equipment won't be much better, but their advertising and marketing will be. Are they serious about this? Matsushita, which markets here under the Panasonic and Quasar names, is serious, and they think they are going to sell over one million computers in 1981 in the U.S. That's a lot more than Radio Shack, Apple and all the rest are figuring to sell.

Casio is thinking big, and with good reason. They have done a fantastic job of taking over a large share of the calculator market in the last three years. I carry at least three Casio calculators, plus a Sharp Talking Clock, around with me most of the time. I think Sharp will be here soon with a computer, too. They are going hot and heavy in Europe expanding their production to where they can be very competitive. Casio says they intend to be the biggest firm in the business here in two years.

As more and more Japanese firms dig their toes in at the starting line for the race across America, I'm watching to see what our American firms will do to counter this invasion. So far this year we've seen entries from Quasar, Panasonic and Casio, as well as from NEC, with Sharp and Hitachi headed this way. A reading of our trade literature makes it apparent that these firms are very serious about taking over and making microcomputers as much of a Japanese preserve as are small cars, stereo systems, television sets, CB radios, amateur radios and most of the other high-technology big-bucks consumer industries.

Surely our American microcomputer firms are aware of what is happening and are preparing for the battle? This is not the case. In fact, as I visit the American firms, I see a never-never-land blindness to what is going on. Wishful thinking and a euphoria resulting from success have virtually wiped out reality.

How can you expect people who are making fabulous salaries, working in lavish offices in multi-building complexes—all generated in the last year or two—surrounded by minions anxious to please and fearful of sounding a disturbing tone to have any perspective? Most of these unfortunates have built up a dreamworld around themselves and their accomplices. It was in such a world that Processor Tech self-destructed. This same protective buffer of unreality helped Imsai to disintegrate. You can be sure that not one of the new tycoons will have the time to read this, much less think seriously about it.

Thus, I see our American industry as being very fragile, built on ever-increasing growth, but virtually blind to what is going on outside of the beautiful executive suites in Silicon Valley as viewed through their tinted picture windows facing on automatically watered grass lawns. I see this as a scenario for disaster as the hard-working and advertising-wise Japanese start moving in.

At first, our industry will try to refuse to acknowledge that the Japanese really exist or are any serious threat. By the time they do begin to see what is happening, it will be too late, and they will be scrambling to see what, if anything, can be salvaged from the ruin. We'll see more of the midnight back-door deals as the formerly anointed back up rented trucks to grab as much as they can from assembly lines before the sheriff puts on the locks.

In addition to this number-one blindness, which I expect the Japanese to exploit, there is one other serious, and perhaps even terminal, weakness in the American microcomputer industry. This is the same weakness that has helped wipe out our car industry—pay scales.

Yes, I know all about the unresponsiveness of the Detroit moguls to the desire for small cars. But while it is unpopular to give the businessmen of Detroit credit for having any brains, the fact is that these gentlemen were well aware of the American need for economy cars. But Detroit had a problem, an insoluble problem: They could not compete against Germany and Japan in building small cars because the American auto workers were getting double the average American wage, while the foreign workers were making about the same wages as Americans.

If Detroit could have started over and been able to pay normal American wages, I would not be driving an RX-7 Mazda and a Datsun 280Z, nor would I be considering a Rover 3500 as a new-car purchase. Japan moved in gradually, keeping their costs down by paying normal wages to their people, keeping executive overhead low and automating in every way possible. I'm sure that our auto unions will fight for their double the average American wages until the last car company goes down the tubes. We've seen this same mentality sinking England for several years with no relief in sight.

I realized that American productivity has been very low compared to other countries and is dropping further behind, but I hadn't realized how much the wage scales forced on the auto industry by the powerful unions had made the industry unable to compete with foreign auto makers, thus forcing the American firms to keep making the larger gas guzzlers, which foreign firms were not. Our firms made the only product they could sell and then backed this up with their massive advertising system. Now that is all falling apart.

How does this apply to the computer industry? We have the same pattern for the Japanese to exploit—excessive salaries, particularly in Silicon Valley, where these astounding figures are almost considered normal, an incredible


kilobaud
MICROCOMPUTING T.M.

lack of automation in the factories and tremendous overhead caused by armies of managers and executives. These excesses, made possible by the 300–400 percent growth per year of the industry, are the seeds of its destruction. The Japanese run a lean and mean ship, so they can come in and stomp us with prices we can't match.

All those executives and opulence add substantially to the cost of our computers, adding far more than the small import duties and shipping costs from Japan. I predict that unless our major firms get out of Silicon Valley and into the low-rent districts and pay average American wages to the few people needed in much more automated factories, we'll see 90 percent of the microcomputers coming from Japan within five years.

Japan took over the ham equipment, stereo, watch, calculator and radio markets. Now we are seeing technology coming from Japan, instead of from the U.S.

Despite this foreign influx, life goes on happily in Silicon Valley, with not even a ripple of concern or reaction from our industry. Radio Shack is in a lower-rent district, but they need to pay attention to their corporate overhead and lack of automation in their plants. They also need some fast work on software support of their products and advertising approach. They *do* have the stores and a two-year head start, so it would be a shame to see them blow it now.

Japan already has some advantages over us by virtue of their lower-cost memory and microprocessor chips. They pulled this off by out-automating our chip firms, thus cutting down the labor costs involved in making the chips. They also have a substantial advantage over us in employee efficiency, oddly enough, using American know-how. You may have read some of the spate of recent articles on how the Japanese brought an American named Demming over to Japan and applied his ideas on a national scale to improve their productivity and attention to quality control.

The Japanese have well-developed Quality Circles to attack every hint of a defect in the quality of their products or service. They worry about how many hours it takes for a computer to be fixed for the customer, the quality of the package it is sold in, the advertising and the brochures. Every detail of business is relentlessly pursued for quality, and the result is superb products, superb advertising and customer enthusiasm and confidence. I *know* when I buy a Casio or Sony product that it is going to be first class in every way and that I am not ever going to have hassles. I wish I could say the same for some of the American firms; my files are filling with bitter customer complaints about their quality and disregard of the customer.

It remains to be seen whether the microcomputer industry can learn from the Detroit debacle and cut their corporate overhead, automate and set up quality control for all phases of their business before it is too late.

Yes, I am painting a grim picture, but my facts are accurate and not exaggerated. I have visited Apple, Atari, Commodore, Radio Shack, Southwest Tech, Texas Instruments, Ohio Scientific, Polymorphics, The Digital Group, Wavemate, Imsai, Sphere, Mits, Astral, ECD, HP and others that don't come immediately to mind. I don't think anyone else is as intimately familiar with our industry. I've been around right from the beginning, talking with the top people in their plants and at shows.

## UK Report

A letter from A. Zgorelec of Britain's *Personal Computer World* magazine, by far the largest of the UK microcomputing magazines with over 100 pages of paid advertising, mentioned that a recent survey estimated that there are about 60,000 microcomputers in that country. About half of those are PET systems, followed by Apple in second place, Nascom (a British computer, now in receivership) in third place and Tandy (Radio Shack) in fourth, with about 8000 systems sold. Tandy has only about 100 stores in UK, as opposed to over 400 stores selling the PET. PET software is far more available and in better quantity and quality.

I suspect that a quick infusion of Instant Software might turn things around for Tandy in UK, but obviously I'm prejudiced. This would call for a change in company policy, something that takes more than the loss of a few million dollars to bring about. *80-Microcomputing* is well thought of by the Tandy managers, but they are not allowed to recommend the magazine to customers.

Clive Sinclair and his $200 single-board Z-80 computer are doing very well, with sales of over 8000 units in eight months and at least a 90-day back-order situation currently. He's making it happen with full-page color ads in the leading Sunday newspapers. This system was shown at CES in Chicago in June, and the signs are that they will be getting going in a serious way in the U.S.

Other manufacturers of microcomputers in UK are Transam, with their Tritan and Tuscan systems, and Acorn Computer, with their Atom and Research machines (strong in the educational field).

## Do Your Homework

It has come to my attention that a rather large number of you readers have been goofing off, not holding up your end of the system. I will not tolerate laziness in a subscriber. When you sign up for *Microcomputing*, you are doing far more than sending in your crummy $25. You are also expected to do your share of the work.

For instance, as part of your responsibility, you are expected to check out the new books being offered by other publishers (we'll take care of reviewing ours) and write brief reviews, pulling no punches about how good or how lousy they are.

This also applies to software. If you purchase a good piece of software, let us know. If you find you've been screwed, level with us. We all will benefit from the news, although *Microcomputing* may lose an advertiser or two. We even want to know your experiences with Instant Software.

While we are able to check out a lot of the new hardware gadgets in our lab, it is impossible for us to do everything. There's just too much. So when you get a newly released product, please send in a log of your experiences with it so we can pass the information along to the rest of the readers. We want to know how it works, what problems you had and how responsive the manufacturer and dealer were.

You may work out some programming routines which would benefit others. If so, think of *Microcomputing* as a way to pass along the information. You'll reach the most people that way. . .by a wide margin. We're all in this together, so the more we help each other, the better off we'll all be.

Send your reports to Software Reports, *Microcomputing*, Peterborough, NH 03458.

You are also expected to work hard to recruit new subscribers, to put maximum pressure on manufacturers and dealers to advertise, to loyally buy advertised products, keeping your readership of the magazine no secret in the process, to respond enthusiastically to the reader service card each month and do all you can to help support the system of your choice.

If a friend has developed a gadget or worked out a program of interest, get him to write it up for *Microcomputing*. If you find a computer store that does not have the magazine on display, let us know immediately so we can correct this terrible state of affairs.

Send me newspaper or magazine articles about microcomputers. I don't want to miss anything, but obviously I can't read every publication there is, so I have to count on *you*.

Okay, now get busy.

## Provide a "HELP" Command

It should not be news to programmers that many of the users of computers are not experts. This should be taken into consideration when programs are written. All programs should provide an explanation of how to use the program by typing HELP.

Many times I find myself faced with a computer program that does not respond to the normal approaches. I really hate that. I don't like to be exposed, even to myself, for the dummy that I can be.

As we get more into business programs, we will have to remember to make them simple to use for someone who has walked up to the computer for the first time. It is not necessary to force experienced people to go through a lot of explanatory stuff every time they want to use the program, but a simple explanation should be available on demand.

Programs submitted for publication and distribution by Instant Software will be trending in this direction. So take note, programmers.

## Conversions Requested

What is more frustrating than to turn the page of a new issue of *Microcomputing* and find just the program you need, only it has been written for use on some other system!

If you are into programming, this will be a hurdle, not a brick wall, and you'll start keying in the code and watching for any commands you know will not perk through your own system. Some systems conversions are more demanding than others, but, in general, there are few programs that will not bend to an iron will. Of course, when it comes to graphics, you're going to have to start pretty much from scratch to generate them with your system. That's where the hard work comes in.

Once you've managed to adapt the program to your system, why be stingy about it? Run off a copy of the new program and send it in to *Microcomputing* for possible publication (paid) to help others who are up against the problem but don't have the time (or smarts) to surmount it. We'd appreciate a cassette copy, too. If you are running a disk, send it in and we'll get it back to you later. This will enable us to check out your version of the program in our lab and possibly print out a clean listing for publication.

This will help readers with smaller-circulation systems build up their program libraries.

## Toes Stepped On

A recent "survey" in one of the newsprint throwaways presented a rather biased report with regard to Instant Software that deserves comment.

The report observed that zero percent of Instant Software is written in-house. This is not quite true. While ISI does not hire programmers to write programs, there are still quite a number of programmers on the staff whose efforts are added to many of the programs and program packages that are released. The object is to make every program as good as it can be and an outstanding value. Thus, the staff often adds routines to programs to enhance them, as do associate editors, who work on contract. The program authors are generally consulted as to any changes in their works, so little of this will come as a surprise to them.

Most of the programs published by ISI are, indeed, volunteered by the programmers. But there are some that are solicited from creditable programmers, and there is some serious discussion about getting some in-house programmers to write a few needed types of programs that have not yet been volunteered or have not been received through solicitations.

The report stated that program evaluation time at ISI was two to three months. About 75 percent of the rejections are made within the first few days, and many programmers have received this bad news within a week of sending in their programs. Once a program gets by the preliminary screening, it is admittedly a lengthy process. But this is to the advantage of the programmer and to the customers. Programs that appear to have promise are sent out to associate editors for evaluation and possible improvement. Only about 25 percent of the programs are rejected after this more extensive evaluation, so the preliminary check appears to be valid and effective.

One of the major differences between ISI and many (if not most) other software publishers is the quality of the finished product. If a program makes it through the lengthy process at ISI, it *is* a good program. Customers and dealers have found that they can depend upon every ISI package to be top-notch and a very good value. Yes, it does take longer to make sure the quality is always good, but this is of critical importance in the long run. Firms that make quick decisions have a very poor quality record.

The documentation for ISI is mostly done in-house. Obviously, the better the material from the author, the faster the program can be produced. ISI has a complete editorial staff to write and edit the documentation, set it in type, paste it up for printing, shoot the negatives for printing, etc. Many of the instruction booklets are even printed in the ISI pressroom. The volume of orders has reached the point where the in-house presses can't handle them, so some instruction booklets are printed by outside printers. (The developing plans for a new ISI building include a much larger printing department.)

The biggest difference between software publishers is in marketing. Small firms have trouble getting the interest of most dealers, since the bookkeeping involved soon becomes prohibitive. Thus, a firm such as Instant Software, with over a thousand released programs from which to choose, greatly simplifies the display and bookkeeping problems. Then, too, ISI reps visit most stores once a month to help them with their displays, to acquaint the salespeople with the new releases, to pick up slow-moving packages or to look into any problems. It is this network of reps that has enabled ISI to reach more stores—well over 400 stores worldwide at present—than any other producer of software.

**Programmers will have to be idiots to bother to write programs for sale if they get little out of it.**

The royalties on ISI programs are normally 20 percent of the ISI gross. If the program is sold by mail order, then the author receives 20 percent of the retail price. If a program sells to a store at 33 percent discount, the author gets 20 percent of that receipt. About 90 percent of the program sales at present are via dealers. This percentage seems to be increasing as the dealer network grows.

Articles about software publishing would be of more value if they were better researched. If a programmer just goes by the meager facts presented in such a survey, he could lose thousands of dollars in royalties by being conned into going with a firm with fast action and small distribution. And at least one of the firms referenced in the report has been fleecing programmers, despite the implied seal of approval given by the publishing of the survey.

## Program Theft

I recently received through the mail a Hayden software catalog, accompanied by a letter that offered a ten-day free examination of the programs. I'm reasonably honest, but I'm not sure I could withstand *that* kind of temptation. I suspect that Hayden is so used to selling books this way that they went right ahead and used the same approach for their computer programs.

While publishing books and computer programs have many similarities, there are some important differences. A book is very difficult and expensive to copy, so ten-day free offers are a valid selling method. Computer programs, which are usually more expensive than books, are so simple to copy that it is pathetic. Free examination offers make no sense in this field. Most of us are honest as long as you don't tempt us too far. But it is so simple to send away for the whole catalog of Hayden programs, dump a copy on a cassette and then return them that I doubt that this sort of nonsense will continue.

That brings me to the thinly disguised services aimed right at program copiers. A chap recently started up a "program library" service that would lend computer programs for a fraction of the regular sale price. Needless to say, Instant Software, which was in his catalog, is preparing a suit to stop this sort of theft.

I've had several letters telling me of clubs that are allowing, or even encouraging, the copying of programs. I hope that these clubs are incorporated, because I'd hate to have to sue every member of the club instead of just the incorporated club itself. Where there is no corporation, every member is liable for the actions of the club. The usual practice is to sue *all* members and then really go after those with money.

It is only by pursuing these thefts of programs that the industry will be able to protect the authors of programs. And it is only by protecting the ability to pay good royalties that we shall have the programs that will allow microcomputers to grow in their sales. To a great extent, much of the future of the whole industry rests on the ability of software firms to support the systems. Programmers will have to be idiots to bother to write programs for sale if they get little out of it.

While all this makes sense to everyone, people continue to run off disk copies of dozens or even hundreds of programs at the club, so why pass it up? Well, I can't speak for the other firms in the business, but if just one person in your club spills the beans to ISI, he stands a good chance of making out rather well. We offer a $10,000 reward for such information and are very serious about it. We might well collect quite a bit more than that in damages.

Give programmers a break and put a stop to program theft wherever you see it being condoned. I think you'll find very few computer stores that will allow this anymore. There is just far too much for them to lose.

Prices reflect distribution on 8" single density diskettes. If a format is requested which requires additional diskettes, a surcharge of $8. per additional diskette will be added. A surcharge of $25 will be added for software on CSSN format DC 300XL cartridges. Media charge for 5440 disk is $100.

# No. 16: Take a byte.

*Genuine CP/M for Apple II Available now!*

**All Lifeboat programs require CP/M, unless otherwise stated.**
*Software for most popular 8080/Z80 computer disk systems*

Software with Manual / Manual Alone

☐ **CP/M* FLOPPY DISK OPERATING SYSTEM**—Digital Research's operating system configured for many popular micro-computers and disk systems:

| System | Version | Price |
|---|---|---|
| New Apple II* SoftCard* with Z80 Microsoft BASIC version 5 with high resolution graphics | 2.x | 350/25 ◇ |
| North Star Single Density | 1.4 | 145/25 Ⓥ |
| North Star Double Density | 1.4 | 145/25 |
| North Star Single Density | 2.x | 170/25 |
| North Star Double/Quad | 2.x | 170/25 |
| Durango F-85 | 2.x | 170/25 |
| iCOM Micro-Disk 2411 | 1.4 | 145/25 |
| iCOM 3712 | 1.4 | 170/25 Ⓥ * |
| iCOM 3812 | 1.4 | 170/25 * |
| Mits 3202/Altair 8800 | 1.4 | 145/25 |
| Heath H8 + H17 | 1.4 | 145/25 Ⓜ |
| Heath H89 | 1.4 | 145/25 Ⓜ |
| Heath H89 by Magnolia | 1.4 | 250/25 ◇ |
| Heath H89 by Magnolia | 2.x | 300/25 ◇ |
| Onyx C8001 | 2.x | 300/25 ◇ |
| Ohio Scientific C3 | 2.x | 200/25 |
| TRS-80 Model I | 1.4 | 145/25 Ⓜ |
| TRS-80 Model II | 2.x | 170/25 |
| TRS-80 Model II + Corvus | 2.x | 250/25 |
| Processor Technology Helios II | 1.4 | 145/25 |
| Cromemco System 3 | 1.4 | 145/25 |
| Intel MDS Single Density | 1.4 | 145/25 |
| Intel MDS Single Density | 2.x | 170/25 |
| Micropolis Mod I | 1.4 | 145/25 Ⓥ |
| Micropolis Mod II | 1.4 | 145/25 Ⓥ |

The following configurations are scheduled for release soon:

| | | |
|---|---|---|
| North Star Double/Quad + Corvus | 2.x | 250/25 |
| North Star Horizon HD-1 | 2.x | 250/25 |
| Ohio Scientific C3-C | 2.x | 250/25 |
| Micropolis Mod II | 2.x | 200/25 |
| Mostek MDX STD Bus System | 2.x | 350/25 ** |
| iCOM 3812 | 2.x | 225/25 * |
| iCOM 4511/Pertec D3000 | 2.x | 375/25 *+ |

*Software consists of the operating system, text editor, assembler, debugger and other utilities for file management and system maintenance. Complete set of Digital Research's documentation and additional implementation notes included. Systems marked * and ** include firmware on 2708 and 2716. Systems marked + include 5440 media charge. Systems marked Ⓜ require the special Ⓜ versions of software in this catalog. Systems marked Ⓥ have minor variants available to suit console interface of system. Call or write for full list of options. ◇ includes hardware addition to allow our standard versions of software to run under it.*

☐ **Z80 DEVELOPMENT PACKAGE**—Consists Ⓜ Ⓩ of: (1) disk file line editor, with global inter and intra-line facilities; (2) Z80 relocating assembler, Zilog/Mostek mnemonics, conditional assembly and cross reference table capabilities; (3) linking loader producing absolute Intel hex disk file .......... **$95/$20**

☐ **ZDT**—Z80 Monitor Debugger to break and Ⓜ Ⓩ examine registers with standard Zilog/Mostek mnemonic disassembly displays. $35 when ordered with Z80 Development Package .......... **$50/$10**

### AVOCET SYSTEMS

☐ **XASM-68**—Non-macro cross-assembler with nested conditionals and full range of pseudo operations. Assembles from standard Motorola MC6800 mnemonics to Intel hex ... **$200/$25**

☐ **XASM-65**—As XASM-68 for MOS Technology MCS-6500 series mnemonics ...... **$200/$25**

☐ **XASM-48**—As XASM-68 for Intel MCS-48 and UPI-41 families .......... **$200/$25**

☐ **XASM-18**—As XASM-68 for RCA 1802 .......... **$200/$25**

☐ **DISTEL**—Disk based disassembler to Intel 8080 or TDL/Xitan Z80 source code, listing and cross reference files, Intel or TDL/Xitan pseudo ops optional. Runs on 8080 ........ **$65/$10**

☐ **DISILOG**—As DISTEL to Zilog/Mostek Ⓜ Ⓩ mnemonic files .......... **$65/$10**

Software with Manual / Manual Alone

☐ **SMAL/80** Structured Macro Assembler Ⓧ Language—Package of powerful general purpose text macro processor and SMAL structured language compiler. SMAL is an assembler language with IF-THEN-ELSE, LOOP-REPEAT-WHILE, DO-END, BEGIN-END constructs .......... **$75/$15**

### PHOENIX SOFTWARE ASSOCIATES

☐ **PASM***—Z80 macro assembler, Intel/TDL Ⓩ mnemonics. Generates Intel hex format or relocatable code in either TDL Object Module format or PSA Relocatable Binary Module format. Supports text insertion, conditional branching within macros, recursive macro calls and parameter passing. .......... **$129/$25**

☐ **EDIT**—Character oriented text file editor. In- Ⓩ cludes macro definition capabilities. Handles insertion, deletion, searching, block move, etc. for files of any length. Does not require a CRT. .......... **$129/$25**

☐ **PLINK***—Two pass **disk-to-disk linkage edi-** Ⓩ **tor/loader** which can produce re-entrant, ROMable code. Can link programs that are larger than available memory for execution targeted on another machine. Full library capabilities. Input can be PSA Relocatable Binary Module, TDL Object Module or **Microsoft REL files**. Output can be a COM file, Intel hex file, TDL Object Module or PSA Relocatable file. .......... **$129/$25**

☐ **BUG*** and **μBUG***—Z80 interactive machine Ⓩ level debugging tools for program development. BUG has full symbolic trace and interactive assembly (mnemonics compatible with PASM). Dynamic breakpoints and conditional traps while tracing (even through ROM!). μBUG is a subset of BUG and is used in memory limited situations .......... **$129/$25**

### DIGITAL RESEARCH

☐ **MP/M**—Installed for single density MDS-800. Multi-processing derivative of the CP/M operating system. Manual includes CP/M2 documentation .......... **$300/$50**

☐ **MAC**-8080—Macro assembler. Full Intel Ⓢ macro definitions. Pseudo Ops include RPC, IRP, REPT, TITLE, PAGE, and MACLIB. Produces absolute hex output plus symbol table file for use by SID and ZSID (see below) **$120/$15**

☐ **SID**-8080—Symbolic debugger. Full trace, Ⓢ pass count and breakpoint program testing. Has backtrace and histogram utilities. When used with MAC, provides full symbolic display of memory labels and equated values . **$105/$15**

☐ **ZSID**-Z80—Symbolic debugger with all fea- Ⓢ Ⓩ tures of SID .......... **$130/$15**

☐ **TEX**—Text output formatter to create paginat- Ⓢ ed, page-numbered and justified copy. Output can be directed to printer or disk ... **$105/$15**

☐ **DESPOOL**—Utility program to permit simulta- Ⓢ neous printing from text files while executing other programs .......... **$80/$10**

☐ **tiny C**—Interactive interpretive system for Ⓧ teaching structured programming techniques. Manual includes full source listings . **$105/$50**

☐ **BDS C COMPILER**—Supports most features Ⓜ Ⓘ of language, including Structures, Arrays, Pointers, recursive function evaluation, overlays. Includes linking loader, library manager, and library containing general purpose, file I/O, and floating point functions. Lacks initializers, statics, floats and longs. Documentation includes "The C PROGRAMMING LANGUAGE" by Kernighan and Ritchie .......... **$145/$25**

☐ **WHITESMITHS C COMPILER**—The ultimate Ⓛ Ⓧ in systems software tools. Produces faster code than a pseudo-code Pascal with more extensive facilities. Conforms to the full UNIX* Version 7 C language, described by Kernighan and Ritchie, and makes available over 75 functions for performing I/O, string manipulation and storage allocation. Linkable to Microsoft REL files. Requires 60K CP/M ..... **$630/$30**

### MICROSOFT

☐ **BASIC-80**—Disk Extended BASIC, ANSI Ⓛ Ⓜ compatible with long variable names, WHILE/WEND, chaining, variable length file records .......... **$325/$25**

☐ **BASIC COMPILER**—Language compatible Ⓛ Ⓜ with BASIC-80 and 3-10 times faster execution. Produces standard Microsoft relocatable binary output. Includes MACRO-80. Also linkable to FORTRAN-80 or COBOL-80 code modules .......... **$350/$25**

Software with Manual / Manual Alone

☐ **FORTRAN-80**—ANSI 66 (except for COM- Ⓛ Ⓜ PLEX) plus many extensions. Includes relocatable object compiler, linking loader, library with manager. Also includes MACRO-80 (see below) .......... **$425/$25**

☐ **COBOL-80**—Level 1 ANSI '74 standard Ⓛ Ⓜ COBOL plus most of Level 2. Full sequential, relative, and indexed file support with variable file names. STRING, UNSTRING, COMPUTE, VARYING/UNTIL, EXTEND, CALL, COPY, SEARCH, 3-dimensional arrays, compound and abbreviated conditions, nested IF. Powerful interactive screen-handling extensions. Includes compatible assembler, linking loader, and relocatable library manager as described under MACRO-80 .......... **$700/$25**

☐ **MACRO-80**—8080/Z80 Macro Assembler. Ⓛ Ⓜ Intel and Zilog mnemonics supported. Relocatable linkable output. Loader, Library Manager and Cross Reference List utilities included .......... **$149/$15**

☐ **muSIMP/muMATH**—muSIMP is a high level Ⓛ programming language suitable for symbolic and semi-numerical processing. Implemented using a fast and efficient interpreter requiring only 7K bytes of machine code. muMATH is a package of programs written in muSIMP. The package performs sophisticated mathematical functions. Keeps track of up to 611 digits. Performs matrix operations on arrays: transpose, multiply, divide, inverse and other integer powers. Logarithmic, exponential, trigonometric simplification and transformation, symbolic differentiation with partial derivatives, symbolic integration of definite and indefinite integrals. Requires 40K CP/M. .......... **$250/$20**

☐ **muLISP-79**—Microcomputer implementation Ⓛ of LISP. The interpreter resides in only 7K bytes of memory yet includes 83 LISP functions. Has infinite precision integer arithmetic expressed in any radix from 2 to 36. muLISP-79 includes complete trace facility and a library of useful functions and entertaining sample programs. .......... **$200/$15**

☐ **XMACRO-86**—8086 cross assembler. All Ⓛ Macro and utility features of MACRO-80 package. Mnemonics slightly modified from Intel ASM86. Compatibility data sheet available .......... **$275/$25**

☐ **EDIT-80**—Very fast random access text editor Ⓛ for text with or without line numbers. Global and intra-line commands supported. File compare utility included. .......... **$89/$15**

☐ **PASCAL/M***—Compiles enhanced Standard Ⓧ Pascal to compressed efficient Pcode. Totally CP/M compatible. Random access files. Both 16 and 32-bit Integers. Runtime error recovery. Convenient STRINGs. OTHERWISE clause on CASE. Comprehensive manual (90 pp. indexed). SEGMENT provides overlay structure. INPORT, OUTPORT and untyped files for arbitrary I/O. Requires 56K CP/M. Specify 1) 8080 CP/M, 2) Z80 CP/M, or 3) Cromemco CDOS. .......... **$175/$20**

☐ **PASCAL/Z**—Z80 native code PASCAL com- Ⓘ Ⓩ piler. Produces optimized, ROMable re-entrant code. All interfacing to CP/M is through the support library. The package includes compiler, relocating assembler and linker, and source for all library modules. Variant records, strings and direct I/O are supported. Requires 56K CP/M .......... **$395/$25**

☐ **PASCAL/MT**—Subset of standard PASCAL. Ⓜ Ⓧ Generates ROMable 8080 machine code. Symbolic debugger included. Supports interrupt procedures, CP/M file I/O and assembly language interface. Real variables can be BCD, software floating point, or AMD 9511 hardware floating point. Includes strings enumerations and record data types. Manual explains BASIC to PASCAL conversion. Requires 32K .. **$250/$30**

☐ **APL/V80**—Concise and powerful language for Ⓛ Ⓩ application software development. Complex programming problems are reduced to simple expressions in APL. Features include up to 27K active workspace, shared variables, arrays of up to 8 dimensions, disk workspace and copy object library. The system also supports auxiliary processors for interfacing I/O ports. Requires 48K CP/M and serial APL printing terminal or CRT .......... **$500/$30**

☐ **ALGOL-60**—Powerful block-structured lan- Ⓩ guage compiler featuring economical run-time dynamic allocation of memory. Very compact (24K total RAM) system implementing almost all Algol 60 report features plus many powerful extensions including string handling direct disk address I/O etc. .......... **$199/$20**

☐ **CBASIC-2** Disk Extended BASIC—Non- Ⓜ interactive BASIC with pseudo-code compiler and run-time interpreter. Supports full file control, chaining, integer and extended precision variables, etc. .......... **$120/$15**

### MICRO FOCUS

☐ **STANDARD CIS COBOL**—ANSI '74 COBOL Ⓛ standard compiler fully validated by U.S. Navy tests to ANSI level 1. Supports many features to level 2 including dynamic loading of COBOL modules and a full ISAM file facility. Also, program segmentation, interactive debug and powerful interactive extensions to support protected and unprotected CRT screen formatting from COBOL programs used with any dumb terminal .......... **$850/$50**

Software with Manual / Manual Alone

☐ **FORMS 2**—CRT screen editor. Output is Ⓛ COBOL data descriptions for copying into CIS COBOL programs. Automatically creates a query and update program of indexed files using CRT protected and unprotected screen formats. No programming experience needed. Output program directly compiled by STANDARD CIS COBOL .......... **$200/$20**

### EIDOS SYSTEMS

☐ **KISS**—Keyed Index Sequential Search. Offers Ⓛ complete Multi-Keyed Index Sequential and Direct Access file management. Includes built-in utility functions for 16 or 32 bit arithmetic, string/integer conversion and string compare. Delivered as a relocatable linkable module in Microsoft format for use with FORTRAN-80 or COBOL-80, etc. .......... **$335/$23**

☐ **KBASIC**—Microsoft Disk Extended BASIC Ⓛ version 4.51 integrated by implementation of nine additional commands in language. Package includes KISS. REL as described above, and a sample mail list program. .... **$585/$45**
To licensed users of Microsoft BASIC-80 (MBASIC) .......... **$435/$45**

☐ **XYBASIC** Interactive Process Control BASIC—Full disk BASIC features plus unique commands to handle byte rotate and shift and to test and set bits. Available in several versions:

| | |
|---|---|
| Integer ROM squared | **$350/$25** |
| Integer CP/M | **$350/$25** |
| Extended ROM squared | **$450/$25** |
| Extended CP/M | **$450/$25** |
| Extended Disk CP/M | **$550/$25** |
| Integer CP/M Run Time Compiler | **$350/$25** |
| Extended CP/M Run Time Compiler | **$450/$25** |

☐ **RECLAIM**—A utility to validate media under CP/M. Program tests a diskette or hard disk surface for errors, reserving the imperfections in invisible files, and permitting continued usage of the remainder. Essential for any hard disk. Requires CP/M version 2. ...... **$80/$5**

☐ **BASIC UTILITY DISK**—Consists of: (1) Ⓜ CRUNCH-14— Compacting utility to reduce the size and increase the speed of programs in Microsoft BASIC 4.51, BASIC-80 and TRS-80 BASIC. (2) DPFUN—Double precision subroutines for computing nineteen transcendental functions including square root, natural log, log base 10, sine, arc sine, hyperbolic sine, hyperbolic arc sine, etc. Furnished in source on diskette and documentation .......... **$50/$35**

☐ **STRING/80**—Character string handling plus routines for direct CP/M BDOS calls from FORTRAN and other compatible Microsoft languages. The utility library contains routines that enable programs to chain to a COM file, retrieve command line parameters and search file directories with full wild card facilities. Supplied as linkable modules in Microsoft format. **$95/$20**

☐ **STRING/80** source code available separately— .......... **$295/NA**

☐ **THE STRING BIT**—FORTRAN character Ⓜ string handling. Routines to find, fill, pack, move, separate, concatenate and compare character strings. This package completely eliminates the problems associated with character string handling in FORTRAN. Supplied with source .......... **$65/$15**

☐ **VSORT**—Versatile sort/merge system for fixed Ⓜ length records with fixed or variable length fields. VSORT can be used as a stand-alone package or loaded and called as a subroutine from CBASIC-2. When used as a subroutine, VSORT maximizes the use of buffer space by saving the TPA on disk and restoring it on completion of sorting. Records may be up to 255 bytes long with a maximum of 5 fields. Upper/lower case translation and numeric fields supported. .......... **$175/$20**

☐ **CPM/374X**—Has full range of functions to create or re-name an IBM 3741 volume, display directory information and edit the data set contents. Provides full file transfer facilities between 3741 volume data sets and CP/M files .......... **$195/$10**

*Coming Soon*

**CPAids***

☐ **MASTER TAX**—Professional tax preparation Ⓛ † program. Prepares schedules A, B, C, D, E, F, G, R/RP, SE, TC, ES and forms 2106, 2119, 2210, 3468, 3903, 2441, 4625, 4726, 4797, 4972, 5695 and 6521. Printing can be on readily available, pre-printed continuous forms, on overlays, or on computer generated, IRS approved forms. Maintains client history files and is interactive with CPAids GENERAL LEDGER II (see below) .......... **$995/$30**

☐ **STANDARD TAX**—As above for schedules A, † B, C, D, E, G, R/RP, SE, TC and forms 2106 and 2441. Also, does not maintain client history files .......... **$495/$30**

☐ **GENERAL LEDGER II**—Designed for CPA's. † Stores complete 12 month detailed history of transactions. Generates financial statements, depreciation, loan amortizations, journals, trial balances, statements of changes in financial position, and compilation letters. Includes payroll system with automatic posting to general ledger. Prints payroll register, W2's and payroll checks. .......... **$450/$30**


**Lifeboat Associates,** 1651 Third Avenue, N.Y., N.Y. 10028 **(212) 860-0300** Telex: 220501
**Lifeboat Associates GmbH,** Aegeristr. 35, 6340 Baar **Telefon 042/31 2931**

Robert W. Baker

# PET-POURRI

## Proganal

I recently received an interesting and impressive utility program for the 8K PET from Benson Greene, 210 Fifth Ave., New York, NY 10010. The program is called Proganal and provides two separate functions: It can list a BASIC program with special formatting and cursor character conversion and can generate a detailed analysis of the actual content and structure of the program. This program was extremely interesting to me, since it follows the line of several programs I've worked on in the past.

The program listings produced by Proganal are formatted with page indications for folding, headings and binding instructions. This is convenient, since the output is designed for five-inch-wide roll paper. The page indications make it easy to fold the listings into a usable form. A separating line after any breaks in the logic sequence of the program is also provided.

All PET graphics and cursor controls are printed in a form that prevents misinterpretation, regardless of any printer limitations. All hidden, secret lines that contain delete graphics are printed in full, and any shifted blanks are identified.

The detailed program analysis includes a wealth of information for the subject BASIC program. It shows all program variables, sorted by name and type, with all line-number references indicated. All BASIC commands, along with their line-number references, are listed. Any functions or operators are tallied, showing the total number of times each is used in the program. Each branch instruction, along with every reference line for the branch "target" line, is listed.

At the end of the analysis, a summary report is printed showing the number of program lines and the range of line numbers used, the total number of BASIC instructions, the total number of variable references, the number of different variable names, the number of branches, the number of logic decisions, a relative complexity figure (shown as %) and the time to process the program file. The analysis output is also formatted and paged in the same format as that used for generating the program listing.

Proganal can handle programs with up to 300 distinct, separate line-number references for each letter of the alphabet, for each of the six types of variables. Currently, the BASIC program that is to be processed must first be saved in listing format and then read as a data file. Benson's cover letter stated that he was working on a newer version that would work with the 2040 disk and overcome this requirement.

For now, you must use a tape file so that the program can take a long time to run. I took

```
 STAPLE                              STAPLE
-----------------------------------------
PROGRAM:                  PROGANAL 3/10/80
-----------------------------------------
CURSOR CODES ARE NOTED IN BRACKETS WITH
NO. OF REPEATS. GRAPHIC CODES ARE ASCII

 CLEAR  =[C]  LEFT    =[L]  RIGHT =[R]
 HOME   =[H]  UP      =[U]  DOWN  =[D]
 DELETE =[T]  REVERSE=[V]   OFF   =[O]
 GRAPHIC=[G]  BLANK   =[B](SHIFTED)

         <<< LISTING >>>

 100 :POKE59468,14
 101 IFPEEK(1022)=8THENPOKE1022,128
 105 GOSUB4100:A=5:B%=25:C%=60
 106 D%=65:E%=80:F%=40:K%=C%
 110 DIMA$(B%,A),B$(B%,A),C$(B%,3)
 111 DIMD$(B%,3),E$(B%,3),F$(B%,2)
 112 DIMG$(D%,1),H$(D%,2),I$(B%,3)
 113 DIMJ$(A),R$(300),V$(15)
 115 FORA=0TOD%:READG$(A,0):NEXT
 121 C4$=CHR$(30):C8$=CHR$(29)
 122 CQ$=CHR$(34)
 125 E$=CHR$(20)+"[L][R][U][D][C][H][V][
O][B]"
 126 F$="TLRUDCHVOB
 128 FORA=1TOE%:I$=I$+"-":NEXT
 130 L$="          ":L1$=L$+L$+L$+L$+L$
 135 L6$="______":K$=L6$+L6$+"_____[D][2
B]"
 140 GOSUB7000:OPEN3,3:OPEN4,U
 150 PRINTL6$L6$L6$L6$
 155 IFIN=0THENPRINT"[V]PUT DATA TAPE IN
 TAPE #1":GOSUB2950:OPEN2,1,0


              - PAGE 1 -
-----------------------------------------
------------------FOLD-------------------
-----------------------------------------
PROGRAM:                  PROGANAL 3/10/80
-----------------------------------------
 160 IFIN=1THENGOSUB1000
 165 J=4:P=-1:O=35:GOSUB3000
 170 IFPT=0ANDU>3THENGOSUB7500
 175 A$="LISTING"
 180 IFJOB=0THENA$="EXPANDED "+A$:IFPT=0
THENM=1
 185 GOSUB3200:GOSUB2990:A$="":TS=TI
 190 :P$="":S$="":V$=""
 195 :W$=":"
 200 :C$=W$
 210 :GET#2,W$:IFW$=CHR$(10)THEN210
 220 IFW$=CHR$(13)THENQ=0:GOTO400
 230 P$=P$+W$
 240 IFW$=CQ$ANDQ=1THENQ=0:W$=":":GOTO29
0
 250 IFW$=CQ$THENQ=1:GOTO195
 260 IFQ=1THENGOSUB600:GOTO200
 270 IFW$<="!"THEN210
 280 IFW$=":"THENGOSUB300
 290 :V$=V$+W$:GOTO200
------------------****-------------------
 300 :NI=NI+1:IFJOB=1THENRETURN
 310 S$=S$+P$:Z=VAL(S$)
 320 :IFVAL(P$)>0THENP$=MID$(L$+P$,LEN(S
TR$(Z))+2):IFF=0THENF=Z
 330 IFLEN(P$)<=F%*MTHEN360
 340 P$=LEFT$(P$,F%*M)+L$+" "+MID$(P$,F%
*M+1):IFLEN(P$)<=E%*MTHEN360
 350 P$=LEFT$(P$,E%*M)+L$+" "+MID$(P$,E%
*M+1)
 360 :PRINT#4,P$:IFLEN(P$)>F%*MTHENO=O+1
:IFLEN(P$)>E%*MTHENO=O+1


              - PAGE 2 -
-----------------------------------------
```

```
-----------------------------------------
PROGRAM:                  PROGANAL 3/10/80
-----------------------------------------
      <<< STRING VARIABLES >>>
 W$   7370 8020 8030 8040 8045 8050
 W$   8060 8110

 X$   500 710 740 810 820 1260 1270
 X$   1300 1320 1330 1350 1520 3600
 X$   8090 8110 8210 8240 8250

 Y$   700 740 750 770 780 1270 1310
 Y$   1330 1350

 Z$   1570 1620 1640 1650 8120 8130
 Z$   8150 8180 8190

------------------****-------------------
      <<< NUMERIC ARRAYS >>>
              -NONE-
------------------****-------------------
      <<< STRING ARRAYS >>>

 A$(  110 2110 2240 6030 6050

 B$(  110 2120 2250 6130 6150

 C$(  110 2140 2270 6230 6250

 D$(  111 2150 2280 6330 6350

 E$(  111 2130 2260 6430 6450

 F$(  111 930 6645
  CUT///                          \\\CUT
-----------------------------------------
PROGRAM:                  PROGANAL 3/10/80
-----------------------------------------
      <<< MNEUMONICS >>>
 24 LEFT$(     26 LEN(         LET
  2 LIST          LOAD         LOG(
 42 MID$(         NEW       35 NEXT
    NOT         3 ON         6 OPEN
 33 OR          2 PEEK(      3 POKE
    POS(      114 PRINT      3 READ
  6 REM         1 RESTORE   57 RETURN
 15 RIGHT$(       RND(         RUN
    SAVE          SGN(         SIN(
    SPC(          SQR(       4 STEP
  1 STOP       27 STR$(        SYS
  3 TAB(          TAN(     192 THEN
 33 TO            USR(      37 VAL(
    VERIFY        WAIT         UNKNOWN

------------------****-------------------
      <<< BRANCH INSTRUCTIONS >>>

   190  400 410
   195  250 520
   200  260 290
   210  210 270
   290  240
   300  280
   320  440
   360  330 340
   400  220
   470  440
   550  420 500
   600  260
   630  610 1280

-----------------------------------------
PROGRAM:                  PROGANAL 3/10/80
-----------------------------------------
      <<< ANALASIS SUMMARY >>>

  488  PROGRAM LINES (1-10090)
 1033  INSTRUCTIONS
 1433  TOTAL VARIABLES
   92  VARIABLE NAMES
  233  BRANCHES
  192  LOGIC DECISIONS
  37%  RELATIVE COMPLEXITY
MINUTES TO PROCESS: 129.783056
```

*Example 1. Sample Proganal output.*

over two hours to analyze 500 program lines. But with unlimited computer time on your own system, why worry about running time?

Proganal was written with an original 8K PET with new ROMs, a 32K Expandapet and an Axiom EX-801 printer connected to a TNW-2000 interface. Several examples of what you can expect as output are shown in Example 1. For more information, send a self-addressed, stamped envelope to Benson Greene. He provides copies on tape with documentation for $10.

## Paper-Mate Command 60

Several months ago, AB Computers of Montgomeryville, PA, announced its Paper-Mate Command 60 word processor package for the PET. Written by Michael Riley, this package is an interesting program at a selling price of only $29. It incorporates full-screen editing with graphics for all 16K or 32K PETs, with tape or disk drives and any printer. It includes many features of the Commodore Word Pro 3 word processor, plus several new features.

Written in BASIC, Paper-Mate is slower than Word Pro 3. Since Word Pro 3 is written in machine language, it can keep up with even the fastest typist when entering text. However, you must watch your typing speed with Paper-Mate, or you will overrun the BASIC input buffer. On the other hand, Paper-Mate can easily be modified or customized to suit your specific taste. The documentation even includes a brief program outline and a list of all BASIC variables used by the program.

For writing text, Paper-Mate has a definable keyboard, so you can use it with either business or graphics machines. You can even use your graphics keyboard in a business keyboard mode, where the top row of keys produces numbers, and the semicolon key produces a period. This makes typing with the graphics keyboard much smoother and faster. Another nice feature is a shift lock for letters only, or you can use the normal keyboard shift lock.

Text-editing features include floating cursor, scroll up or down, page forward or back and repeating insert and delete keys. Text block handling includes transfer, delete, append, save, load and insert. The editing features are not as outstanding as those in Word Pro 3, but they get the job done.

All formatting commands are embedded in the text for complete control, but commands must be entered on a separate line. Paper-Mate's commands include margin control and release, column adjust, nine tab settings, variable line spacing, text justification, text centering and auto print form letters (variable blocks). Files can be linked so that one command prints an entire manuscript. Auto paging, page headers, page numbers, pause at end of page and hyphenation pauses are also included. The hyphenation pause gives you the option of either placing the separating hyphen in a long word that overflows a line or keeping the entire word intact on the next line.

With this word processor, you can use PET graphics, as well as text. It can send any specific ASCII code to the printer. This allows multiple-expanded print on a Commodore 2022/2023 printer, something that Word Pro 3 does not support.

The following examines several features provided by Word Pro 3 but not supported by Paper-Mate, or provided in a limited manner.

1. You cannot display the disk directory if you are using a CBM 2040 disk for file storage. However, you can send any other command to the 2040 disk over the disk command channel.

2. There are no search or replace features.

3. Page numbers must be at the bottom of the page. They cannot be selectively used in headings, text, etc.

4. Only a single page heading is generated, centered on the top of the page. Word Pro 3 allows left, middle and right fields within the heading—each positioned accordingly in the heading line.

5. You cannot combine commands on a single line; each Paper-Mate command must be on a separate line.

6. When editing text, you must use the cursor up/down keys to move from line to line. The cursor left/right keys do not wrap from line to line, as in the Word Pro 3.

The Paper-Mate program is a good, usable word processor for the occasional user, considering its lower cost. However, I still recommend Word Pro 3 for anyone requiring extensive controls and fast response.

## Light Pen Programs

In a recent column I mentioned Quill Software of 2512 Roblar Lane, Santa Clara, CA 95051, as a source of programs for use with the 3G Light Pen. They currently offer six different tapes, each with two programs.

Swords and Sorcery—An adventure game using the light pen to fight trolls, find gold, etc. An extensive version that barely fits in 8K. A separate introduction program uses plenty of graphic effects to explain general tactics.

Darth Vader and Hunt the Wumpus—Locate Darth Vader with your laser light beam or hunt the Wumpuses in a much handier fashion.

Laser Shoot and Light Pen Keyboard—In Laser Shoot, you point at a moving cursor to fire at various targets. It's complete with sound effects as well as graphics. The other program draws a replica of the PET keyboard for use as an excellent light pen demo.

Othello and Owari—Play against the computer using the light pen to select your moves.

Quill Quiz—Multiple-choice quizzes to test your knowledge of states/capitals, Spanish/English, vocabulary and historical dates/events.

Billiards and Hangman—Play billiards using the light pen as the cue. In Hangman, use the light pen to save the victim or watch him climb the scaffold and hang.

Quill Quiz is one of the best quiz-type programs I've seen. Answers are selected by simply pointing the light pen at a flashing box next to the answer. You can even select the number of choices (up to ten). You can also determine the question-answer format. For example, choose the state that matches a given capital, or choose the capital that matches the given state.

This program looks as if it could easily be changed to provide quizzes for new topics. All the data is contained in DATA statements, and the program itself is very straightforward.

Each tape sells for $20, plus $1.50, handling. Be sure to specify whether your system has the old or new ROMs.

## CMS Business Software

Over the past several months I've thoroughly tested a business software package from Chuck Stuart at CMS Software Systems (5115 Menefee Drive, Dallas TX 75227). The series of four program packages—General Ledger, Accounts Payable, Accounts Receivable and Payroll—is structured around the time-tested and proven series of business software systems developed by Osborne and Associates. The programs provided by CMS are designed to fill the need for a comprehensive accounting package for the PET. Each program can either stand alone or be integrated with the others in a total software system, depending on your needs.

Designed with the first-time user in mind, the programs lead the operator through verified data entry, step by step. It is impossible to crash a program due to operator error or invalid data input. Design consistency has been maintained from program to program to greatly increase operator familiarity and confidence.

Documentation—normally a problem for small-systems users—is provided by the comprehensive series of Osborne and Associates user manuals. These three manuals total over 800 pages of detailed step-by-step instructions written at three levels for DP department managers, data entry operators and programmers. A second set of manuals details any operations not covered in the Osborne manuals and any program changes made during conversion to the Commodore system. Each program is provided on disk with complete documentation packaged in a handsome, three-ring binder.

The features of the four packages include:

**General Ledger**

- holds up to 300 accounts.
- accepts up to 3000 transactions per month.
- includes cash disbursements, cash receipts and petty cash journals for simplified data entry.
- maintains account balances for the present month, quarter and year, as well as for three previous quarters and the previous year.
- includes complete financial reports, including trial balance, balance sheet, profit and loss statement, cash receipts journal, cash disbursements journal, petty cash journal and more.
- accepts postings from external sources such as the Accounts Payable, Accounts Receivable and Payroll packages. Price is $295.

**Accounts Payable**

- automatic application of credit and debit memos.
- maintains complete purchase records for up to 200 vendors.
- invoice file accepts up to 400 invoices.
- random-access file organization allows fast, individual, record updating.
- multiple reports provide a complete audit

trail.

- check printing with full invoice detail.
- full invoice aging.
- automatic posting to general ledger. Price is $195.

**Accounts Receivable**

- maintains invoice file for up to 300 invoices.
- accommodates full or partial invoice payments.
- customer file maintains purchase information for up to 1000 customers.
- allows automatic progress billing.
- provides for credit and debit memos, as well as invoices.
- prints individual customer statements.
- automatic posting to General Ledger. Price is $195.

**Payroll**

- maintains monthly, quarterly and yearly cumulative totals for each employee.
- payroll check printing with full deduction and pay details.
- sixteen different reports, including W2 and 941 forms.
- complete job costing option with cumulative totals and overhead calculations.
- random-access file organization for fast updating of individual records.
- automatic posting to General Ledger. Price is $350.

After using the General Ledger package in a real-life application for several months, I can say this is one of the best packages I've tried. The programming includes many built-in features and options too numerous to delineate.

If you have a sound interface, the program can provide an error warning signal or a short beep whenever a key is struck (audio verification). Along with the main programs of the package, several disk utilities are included for disk copying, erasing (scratching) account files or zeroing account totals.

CMS provides an update service that allows you to return your original program disk and receive a copy of the latest version for a small handling charge. If necessary, you'll also receive any utility programs required to update your active disks without disturbing the accounting data.

CMS Software also offers several elaborate game programs for the 16K PET: Baccarat, Backgammon, Blackjack, Checkers, Craps, Cribbage, Go Moku, Othello, Quibic 4, Roulette and Space Invaders. Backgammon and Space Invaders cost $9.95 each; all other programs are $7.95 each.

| Address Hex | Decimal | Purpose |
|---|---|---|
| E000 | 57344 | Initialize editor |
| E003 | 57347 | Get a key, return in A |
| E006 | 57350 | Input a line |
| E009 | 57353 | Print a character in A |
| E00C | 57356 | Interrupt handler |
| E00F | 57359 | Time update & keyboard scan |
| E012 | 57362 | Interrupt exit |
| E015 | 57365 | Clear screen |
| E018 | 57368 | Set text mode, upper/lower case display |
| E01B | 57371 | Set graphic mode, upper case/graphics display |
| E01E | 57374 | Set CRT controller |
| E021 | 57377 | Scroll screen down 1 line |
| E024 | 57380 | Scroll screen up 1 line |
| E027 | 57383 | Scan keyboard |
| E02A | 57386 | Ring bell |
| E02D | 57389 | Set repeat flag |
| E030 | 57392 | Set top left limit of scroll window |
| E033 | 57395 | Set bottom right limit of scroll window |

*Example 2. Screen editor system calls.*

## Low-Cost Software

Russell Grokett, in cooperation with the Jacksonville Area PET Society, has made available a large amount of low-cost software for the PET computer. Included are games, finance, ham radio, astronomy, music, graphics and utility programs. Also being added are four-voice music and visible memory graphics programs. Most programs are available for only $1.50 each, plus a small postage fee. Send an SASE to PET Library, 401 Monument Rd. #123, Jacksonville, FL 32211, for a three-page list of the programs available.

Programmatics Software, 71 Sargent Ave., Providence, RI 02906, has also announced a list of software available for the PET. Many of the game programs use joysticks, and a parts kit will be available to build a simple interface for the PET. Other programs include a Billboard program with one-inch high characters and input editing, controlled scrolling speed, etc.

I just released several programs of my own. If interested in a list of PET programs and/or products, send an SASE to Baker Enterprises, 15 Windsor Dr., Atco, NJ 08004.

## Short Notes

Back in March the premiere of the new variety show "Pink Lady" featured a Commodore PET as a graphic device during a sequence by the musical group Blondie. A real-time spectrum analyzer was used to translate the musical tones onto the screen, displayed as a fluctuating bar graph. As far as anyone knows, this was the television debut of the Commodore PET.

An article in the March 17 issue of *Computerworld* described a specially adapted PET that is being used by a gifted, but spastic, 14-year-old boy in Dublin, Ireland. The young poet suffers from severe athetoid cerebral palsy. He is unable to speak and has almost no control of his movements. With the aid of the PET and specially prepared software, he can now write, select, reread and edit his literary material. Commands to the system are entered using his chin or knees and specially designed switches.

LRC, Inc., of Riverton, WY, has announced a new printer for the PET for under $400. The model 7000+ provides 40 characters per line with a print speed of 1.25 lines per second. An available option provides 64 characters per line. This impact printer uses standard, low-cost roll paper.

One of our readers passed on this word of caution: If you order an Excel printer, be sure the unit has been tested. A printer received back in February apparently had an incorrect ROM set and printed in Kata Kana, Japanese phonetic characters. I should mention, however, that Excel promptly refunded the money for the unit when it was returned.

A list of screen editor system calls for anyone with a new 8016 or 8032 CBM with version 4, disk BASIC is shown in Example 2.

Walter Koetke

# COMPUTER BLACKBOARD

## Dealing with Educational Realities

The person who writes a program almost certainly learns more than the person who uses that program. This is readily apparent regarding the learning of programming skills. It is also equally valid regarding the game, simulation, drill or problem that is being programmed.

Consider, for example, a program that begins by asking a student to think of any country in the world. After asking the student a few geographically-based yes/no questions, the program will correctly identify the country selected by the student. I've observed the use of this program by a third grader who used an atlas before answering many of the questions. The boy was obviously enjoying himself while learning world geography and developing map-reading skills. This was a marvelous use of a well-written program that certainly would have warmed the heart of the person who wrote the program.

Nevertheless, the program author learned a good deal more geography than the third-grade user. The author had to first identify all of the countries in the world. This is a nontrivial, real-time task on which reputable authorities appear to disagree. The author then had to write the yes/no questions that would enable the program to uniquely identify each country. This was followed by the task of creating a programmable procedure that would provide all of the desired interaction. Finally comes writing, typing and debugging the actual program code.

There's an important message in this example. Whenever possible, let the students write their own programs. The microcomputer is a magnificent "what if" machine. Students should always be encouraged to experiment with their ideas. They should never just wonder, "What if I do this or that?" They should do it and experience the consequences. The computer cannot be harmed or even insulted by what is typed by a curious student. What appears to be a major disaster can be completely eliminated by turning the microcomputer off and on again.

This message—that students should be encouraged to write their own programs—is easily lost when you visit your local computer store, watch a demonstration, explore the programs in an educational resource center or even read a magazine. There are over 500 software vendors currently competing to sell you programs. Some of these vendors have a few sound products appropriate for education. Almost none of them have products that can be used while a student is using the microcomputer to write his own program. Thus, there is a heavy commercial emphasis on already written program packages.

I encourage you to examine these packages and buy the better ones. But never lose sight of today's most significant microcomputer application—students writing their own programs.

```
10 REM STUDENT NAMES - VERSION 2
20 CLS : RESTORE : M=0 : PRINT
30 PRINT "PLEASE TYPE YOUR FIRST NAME"; : INPUT N$
40 READ F$, L$
50 IF F$="XXXX" THEN 90
60 IF F$<>N$ THEN 40
70 IF M=0 THEN PRINT "YOUR LAST NAME MUST BE " L$ : M=1 : GOTO 40
80 PRINT TAB(20) "OR " L$ : GOTO 40
90 IF M=0 THEN PRINT "YOU DON'T BELONG IN THIS CLASS."
100 PRINT @976,"PRESS THE C-KEY TO CONTINUE";
110 X$=INKEY$ : IF X$<>"C" THEN 110
120 GOTO 20
300 DATA BARRY, WATERS, TOM, JONES, RHONDA, STEWART
310 DATA TOM, SMITH, ALICE, PETERS, CAROL, ALLEN
320 DATA CAROL, WILSON, BILL, HARRIS, TOM, MCNALLY
330 DATA DAVID, OLSON, DANIEL, SMITH
390 DATA XXXX, XXXX
```

*Listing 2.*

```
PLEASE TYPE YOUR FIRST NAME? RHONDA
YOUR LAST NAME MUST BE STEWART

PLEASE TYPE YOUR FIRST NAME? TOM
YOUR LAST NAME MUST BE JONES
OR SMITH
OR MCNALLY

PLEASE TYPE YOUR FIRST NAME?
```

*Sample run 1.*

Although you may be personally convinced of the validity of the preceding sermonette, the realities of your school situation may not permit its full implementation. After all, a student uses considerably more computer time programming and debugging than another who merely runs an educationally sound program prepared by someone else. Let's take a look at a couple of typical situations that aren't very supportive of each student writing his own program and some alternate solutions.

Consider the elementary school principal who is convinced that computer literacy must be incorporated into the curriculum for each of his 600 students. This principal is also committed to equal opportunities for all of his students. He does not want microcomputers to become the exclusive province of a small group of teachers or students. As you and I nod our heads in agreement with his goals, he throws a curve.

How can his students begin to achieve computer literacy with just one or two microcomputers? After considering several possibilities, the principal requested that a program be written to implement the following plan. He hoped to put a microcomputer in the hall outside his office with the program already loaded and running. One morning he would announce that each first grader should type his

```
10 REM STUDENT NAMES - VERSION 1
20 RESTORE : M=0 : PRINT
30 PRINT "PLEASE TYPE YOUR FIRST NAME"; : INPUT N$
40 READ F$, L$
50 IF F$="XXXX" THEN 90
60 IF F$<>N$ THEN 40
70 IF M=0 THEN PRINT "YOUR LAST NAME MUST BE " L$ : M=1 : GOTO 40
80 PRINT "OR " L$ : GOTO 40
90 IF M=0 THEN PRINT "YOU DON'T BELONG IN THIS CLASS."
100 GOTO 20
300 DATA BARRY, WATERS, TOM, JONES, RHONDA, STEWART
310 DATA TOM, SMITH, ALICE, PETERS, CAROL, ALLEN
320 DATA CAROL, WILSON, BILL, HARRIS, TOM, MCNALLY
330 DATA DAVID, OLSON, DANIEL, SMITH
390 DATA XXXX, XXXX
```

*Listing 1.*

first name on the computer "just to see what happens." He wanted the program to respond by displaying the last name of the student. After two or three days, he would make the same announcement to the second grade, and so forth.

The initial version of a program that met these requirements was written by a fourth grader in another school. See Listing 1 and Sample run 1.

That's all there is to it. By changing the DATA statements, you can make the program apply to your class. You might also add nicknames so the computer can identify students by first name or nickname. For example, if Tom Smith's nickname is Ferd, then the DATA statements would contain both FERD, SMITH and TOM, SMITH. Alternately, the student might enter his full name and have his address displayed or enter his phone number and have his name or date of birth displayed.

Remember to use "XXXX", "XXXX" (now in line 390) as the final two items in the DATA statements, because the program uses this as an indicator that there are no more names in the list. Be careful when entering the DATA lines of this program. If you misspell a student's name, he is going to feel badly when the computer is unable to identify him.

As is the case for virtually all programs, this one can be improved. One improved version for the TRS-80 is illustrated in Listing 2. Note that the changes (lines 20, 80 and 100–120) are all user oriented. They provide a more attractive display and a reduced chance of student confusion. Should you use the program, you'll probably make additional improvements of your own.

Return for a moment to the hallway and the first graders entering their first names. Use of this program accomplished several things. Students and teachers alike learned they could touch a computer without harming themselves or the machine. Student interest was very high. They wondered, "How did it know my name? What else does it do? How does it print so fast?"

As a necessary part of his goal regarding computer literacy, the principal distributed copies of the entire program to the students during the week following their first computer experience. What did they see? They saw a few lines of BASIC that they didn't yet understand, followed by the first and last name of every student in their grade.

Then several important ideas became apparent. The computer uniquely identified Rhonda because there's only one Rhonda in first grade. Tom was given three possibilities for his last name because there are three Toms in the first grade. The students learned that the computer can process data very rapidly. But the computer does not know anything we haven't told it. That's an important idea on the road to computer literacy.

Now consider an altogether different situation. Suppose you are completely convinced that students should write their own programs whenever possible. However, educational realities require that you convince other teachers of a variety of subject areas that the computer will also be useful to them. This is a rather common political situation for teachers in many schools.

```
10 REM   DRILL AND PRACTICE  --  FOR ANY SUBJECT AREA
20 CLS : READ N : DIM W(N)
30 C=0 : FOR I=1 TO N : W(N)=0 : NEXT I
40 PRINT "HOW MANY REVIEW QUESTIONS WOULD YOU LIKE"; : INPUT R
50 FOR K=1 TO R
60 CLS : RESTORE : READ N, Q$
70 P=RND(N) : IF W(P)=1 THEN 70
80 W(P)=1 : T=0
90 FOR I=1 TO P : READ A$, B$ : NEXT I
100 PRINT : PRINT Q$; A$; : INPUT R$
110 IF R$=B$ THEN PRINT "CORRECT" : C=C+1 : GOTO 140
120 IF T=0   THEN PRINT "NO, TRY AGAIN." : T=1 : GOTO 100
130 PRINT "NO, THE ANSWER IS "; B$
140 PRINT @976,"PRESS THE C-KEY TO CONTINUE";
150 X$=INKEY$ : IF X$<>"C" THEN 150
160 NEXT K
170 CLS : PRINT "YOU ANSWERED" C "OF" R "REVIEW QUESTIONS CORRECTLY."
300 DATA 10, WHO IS THE COMPOSER OF
310 DATA YELLOW SUBMARINE, BEATLES
320 DATA SWAN LAKE, TCHAIKOVSKY
330 DATA STARS AND STRIPES FOREVER, SOUSA
340 DATA THE BRANDENBURG CONCERTOS, BACH
350 DATA THE NEW WORLD SYMPHONY, DVORAK
360 DATA 50 WAYS TO LEAVE YOUR LOVER, SIMON
370 DATA CAMELOT, LOWE
380 DATA OKLAHOMA, RODGERS
390 DATA 76 TROMBONES, WILLSON
400 DATA PETER AND THE WOLF, PROKOFIEV
```

*Listing 3.*

```
RUN
HOW MANY REVIEW QUESTIONS WOULD YOU LIKE? 4

WHO IS THE COMPOSER OF THE BRANDENBURG CONCERTOS? BACH
CORRECT

WHO IS THE COMPOSER OF YELLOW SUBMARINE? EAGLES
NO, TRY AGAIN.

WHO IS THE COMPOSER OF YELLOW SUBMARINE? BEATLES
CORRECT

WHO IS THE COMPOSER OF 50 WAYS TO LEAVE YOUR LOVER? SIMON
CORRECT

WHO IS THE COMPOSER OF CAMELOT? SOUSA
NO, TRY AGAIN.

WHO IS THE COMPOSER OF CAMELOT? RODGERS
NO, THE ANSWER IS LOWE


YOU ANSWERED 3 OF 4 REVIEW QUESTIONS CORRECTLY.
READY
```

*Sample run 2.*

Let's examine a single program that can be used in nearly all subject areas, although in this instance the program is adapted to music. (See Listing 3 and Sample run 2.) This program is a no-frills drill and practice exercise, but its versatility just might help convince others that microcomputers are useful. Check last month's article for one appropriate use for this type of program.

Briefly, the program selects a question at random from those in the DATA statements. If the student answers a question correctly, another question is randomly selected. If a question is answered incorrectly, the question is repeated. If the question is answered incorrectly again, the right answer is given and the next question is selected.

There are several aspects of this program worth noting. Although rather brief, it contains several features that might be included in your own drill and practice routines. Line 40 permits the student to enter the number of questions desired. This is helpful, since all students don't require the same amount of review. This feature also makes the program usable within the changing time constraints of the classroom.

Lines 70 and 80 ensure that the randomly selected questions will not be repeated during a single session. The variable T in lines 80 and 120 is used to indicate how many times a question has been incorrectly answered. This can be easily modified to allow additional tries before the correct answer is given.

Lines 130–150 allow the student to proceed at his own rate. A question and the correct answer are only cleared from the screen when the student indicates he is ready to continue.

If you don't understand lines 10 through

```
300 DATA 8, WHICH WORD IS THE NOUN --
310 DATA BILL RAN FAST., BILL
320 DATA THE HORSE WAS TIRED., HORSE
330 DATA A SMART PERSON IS THINKING., PERSON
340 DATA THE RACE WAS CLOSE., RACE
350 DATA CANDY IS SWEET!, CANDY
360 DATA THE PROBLEM IS SOLVED., PROBLEM
370 DATA MY PENCIL IS SHARP., PENCIL
380 DATA COMPUTERS ARE FUN TO USE!, COMPUTERS
```

*Listing 4.*

```
300 DATA 11, TYPE THE MISSING NUMBER
310 DATA 2  4  -  8 , 6
320 DATA 217  -  219  220 , 218
330 DATA 27  30  33  - , 36
340 DATA 5  10  -  20 , 15
350 DATA -  7  10  13 , 4
360 DATA 4  -  12  16 , 8
370 DATA 120  130  - 150 , 140
380 DATA -  88  90  92 , 86
390 DATA 21  28  35  - , 42
400 DATA 27  36  -  54 , 45
410 DATA 15  -  31  39 , 23
```

*Listing 5.*

170, that's OK for now, but you should try to eventually understand them. Line 300 is special since it contains the number of questions and answers (only ten in Listing 3), as well as the common portion of each question. DATA lines from line 310 to the end of the program contain each of the questions and answers that might be selected in the drill.

The most important feature of this program is that it provides you with the ability to completely change not only the questions, but also the subject area and question format by merely retyping the data lines. For example, Listing 4 illustrates data for a drill that might be appropriate for an English teacher, while Listing 5 illustrates data for an elementary school mathematics lesson.

Don't be misled by the brevity of these examples. A 16K microcomputer with just cassette storage can hold hundreds of questions and answers in a single program. A teacher who takes the time to prepare a sizable number of drill questions will provide students with an effective, comprehensive tool for review.

Remember that the student who writes a program learns more than anyone who uses the program. There are, however, many circumstances in which the programs of others can be very helpful. Hopefully the programs in this article will help you address two very common educational realities—serving many students with minimum microcomputer availability and demonstrating to other teachers that the microcomputer can help them in their own subject areas. The second example goes even further in that it demonstrates that teachers with no programming skills can still tailor drills for their students in precisely the fashion they choose.

**Correspondence concerning this column should be addressed to Walter Koetke, Putnam/Northern Westchester BOCES, Yorktown Heights, NY 10598.**

# BOOK REVIEWS

## Optimization Techniques in FORTRAN

Joel L. Sears
Petrocelli Books, Inc.
New York, 1979
90 pages, softcover, $10

The title of this book is a misnomer.

Optimization, taken in its usual context, means to use some scarce resource of a machine less frequently. This can be done through such methods as making the program run faster or requiring less primary storage.

But in most cases, Sears does not deal with faster run times or less memory.

*Programming Tricks in FORTRAN* might have been a more descriptive title.

Sears shows numerous ways to get more out of your compiler than you can with most FORTRAN dialects. Using sample programs and clear explanations, he covers such topics as character manipulations, input/output, internal documentation facilities and subprograms.

While the material is oriented toward FORTRAN, many of the techniques can easily be adapted to other programming languages. To the FORTRAN programmer, this book can be an invaluable aid.

To the non-FORTRAN programmer, this book can help him get a little more out of his translator.

**W.A. Harrison**
**Rolla, MO**

## Digital Experiments, 2nd Edition

Richard E. Gasperini
Hayden Book Co., Inc.
Rochelle Park, NJ, 1978
192 pp., $8.95

This is essentially a workbook that will guide you through 25 hands-on verification experiments. These experiments are designed to acquaint you with the operation of basic ICs and LEDs, and teach how to power them, what the pin connections look like, what kinds of outputs are obtained in response to various inputs and how to connect them.

The experiments progress not only by the sophistication of the circuit but by the amount of guidance given. The initial experiments are highly structured, with few decisions required. Later, you must look up your own pinouts. Finally, even the schematic must be worked out.

In the introduction you are told about a logic lab, which is essentially a breadboard for ICs, power supply, switches for data input, LEDs for level indication and a slow and fast clock.

The book recommends how to buy labs or lab kits, and gives sufficient circuitry and instructions for those who care to design and build their own. It also makes recommendations regarding the acquisition of a logic probe, pulser and other equipment to carry out the experiments in this book. Seventeen different types of ICs and LEDs to be studied in these experiments are listed.

All the data sheets required for these ICs and more are furnished in an extensive appendix. Other appendices furnish data sheets on the LEDs, IC cross-references and sources of ICs.

Gasperini includes everything necessary to get the beginner started. If you choose to build your own lab, as you might after reading the recommendations, you will find not only that you have a better lab at far less cost, but also that your digital education is already well-underway.

The experiments start with the measurement of logic levels. They continue by studying the operation of the 7404 hex inverter, what happens when input leads are left open and troubleshooting. Experiments 3 to 6 cover the operation of the AND, NAND, OR and NOR gates, and 7 covers LEDs. Following this, the book discusses decoders in general, and studies a BCD to decimal decoder. Since multiplexers

are somewhat related, they are treated next. Experiment 10 covers the exclusive-OR as a comparator and as a programmable inverter.

Starting with Experiment 11, you are forced to determine your own pin connections (from the data sheets appendix). The next few experiments study the interconnection, operation and use of various kinds of latches and flip-flops. These are followed by several styles of counters, drivers and displays.

Each of these experiments is well thought out, and you are supplied with definitions, explanations and diagrams. It would be hard to go wrong.

In Experiment 23 the reader is closely guided through the connection and operation of a random access memory. In Experiment 24, you are encouraged, with some guidance, to connect and operate a digital clock, using many of the ICs studied in the previous exercises. The last experiment is totally unstructured.

It is difficult to review this book without mentioning another book by the same author, on much the same subject. *Digital Troubleshooting* contains a great deal of very practical digital theory that would very nicely augment the theory given in *Experiments*. I recommend that you use the books together.

This book, along with its companion, is one of a growing number that start at a beginner's level and progress toward something more complicated. While many attempts fall short, this one does not. The book is easy to read and thoroughly enjoyable to work with, and I highly recommend it.

**Alfred Adler, Ph.D.**
**Tucson, AZ**

---

### Computer Systems Organization and Programming

Harry Katzan, Jr.
Science Research Associates, Inc.
Chicago, IL 1976
Hardcover, 459 pp., $17.95

Many books cover a specific topic in great detail. But sometimes the computerist wants a more general survey of the many aspects of computer systems and programming. *Computer Systems and Organization* is just such a book.

It starts off with an introduction to algorithms, sequential machines and grammars, and progresses to programming languages and computer architecture. It discusses in detail the processing unit, main storage and input/output organization, and covers number systems, complement arithmetic, internal data representation, structures, lists and arrays.

The book provides a good introduction to general machine language. In addition, it covers programming techniques, including string manipulation and list processing operations.

An introduction to computer software surveys assemblers and discusses their relation to monitors and linkage editors. A section on programming techniques introduces macros, presents the relation of chaining, overlaying and common storage to the machine and structured programming and provides a thorough treatment of subprograms.

Two chapters on data management introduce external storage devices, file structures, storage device technology and data management functions, and end with a condensed treatment of data base concepts.

The author also deals with operating systems and their functions. He discusses operating system organization, scheduling, allocation, interrupt control and virtual storage methods.

*Computer System Organization and Programming* is a good survey of organization and programming, and does a good job of presenting the concepts that are essential for additional study. But while the book seldom gets too detailed, you should have a basic knowledge of computers. This is not for neophytes.

It is also an excellent reference book.

**Warren A. Harrison**
**Rolla, MO**

# MICRO QUIZ

*This month, we are pleased to introduce our readers to "Micro Quiz," a new column that will be appearing regularly in the pages of* Kilobaud Microcomputing. *The column appears through the efforts of Marc Brown, director of the New England Computer Science League (NECSL), which administers monthly computer contests for high-school students.*

*Each month, Marc will test* Kilobaud Microcomputing *readers with programming problems and computer questions—varying in degree of difficulty—taken from previously held NECSL contests. Answers will appear in the back pages of the same issue.*

*To help you better understand the source of the questions used in this column, we have included a brief profile of the history and activities of the NECSL.*

Organized in 1978, the New England Computer Science League (NECSL) conducts contests for high-school students and awards prizes to outstanding students and schools on local and regional levels.

NECSL motivates students to study computer topics not covered in their school's curriculum and to pursue classroom topics in depth. It encourages students to exchange intellectual ideas and helps them to improve their computer skills. It has already provided the impetus for some high schools to expand and improve their computing facilities and course offerings.

Contests are held simultaneously at each participating school, and an unlimited number of students from all grade levels may compete at each school. A school's score for each contest is the sum of the scores of its five highest-scoring students. In each contest, students are given short theoretical and applied questions and then a practical programming problem to solve (within the following two days) using their school's computer facilities. After the contest is administered by the faculty advisor, each school's results are returned to the League for tabulation. At the completion of each year's contests, an all-star competition is held.

Short questions are designed to be both easy and difficult. This encourages students from all grade levels to participate. Much of the subject matter is new to students, but the programming problems are designed to be useful, practical real-world applications.

Providing graduated test data is another major appeal of the League. Novices can write programs on simplistic levels to handle the first few test data elements, while experienced programmers can write programs on sophisticated levels to handle the more difficult test data elements.

High schools have chosen to implement NECSL in many different ways. Some schools that do not offer any computer courses have used NECSL as the focal point for their computer club. Some have used the contest materials to supplement existing courses, and some schools have even used NECSL as the outline for an entire course in computer science.

The NECSL idea blossomed under the direction of Marc Brown, currently a graduate student in Computer Science at Brown University in Providence, Rhode Island. In its first year, NECSL operated in an experimental mode only in Rhode Island. It soon became apparent that there was a tremendous need for a computer science league, and the League then expanded into the remaining New England states the next year. Today, participation is open to all schools throughout the entire country.

Currently, a handful of annual programming contests exists throughout the United States. NECSL is unique because its contests cover many aspects of the science, not merely programming, and contests are regularly held each month, not yearly.

*Queries regarding column questions or solutions or the NECSL should be addressed to:*
*Marc Brown*
*NECSL*
*Box 1910*
*Brown University*
*Providence, RI 02912*

Upon completion of the following program, how many different elements of array X will have been accessed?

```
DIM     X(10)
FOR     J=1 TO 6
        READ A
        X(A-J)=J
NEXT    J
DATA    8, 3, 10, 6, 7, 7
```

*answer on page 214*

Edited by Dennis Brisson

# NEW PRODUCTS

## Panasonic's Handheld Computer

The RL-H1000 is a modular handheld computer (HHC) that features a full complement of separate peripheral devices including an input/output interface to attach up to six additional peripherals, an acoustic coupler/telephone modem, cassette interface, video RAM interface for hook-up to a home TV set, a mini printer and RAM and ROM memory expanders. The RL-H1000 accommodates four 16K, 32K, 64K or 128K ROM capsules. Four additional capsules can be added with a ROM expander module. It has 65 re-assignable keys and a keyboard overlay system. A complete ASCII character set, in addition to the "help" key, makes traditional command programming easy. Three user-definable function keys accommodate specific capsule programs, multiple key sequences and specialized terminal functions. A memory key allows entrance into RAM from within any program and automatically selects the proper RAM mode for each program. With the four-direction window control, you can drive anywhere within any program or memory file.

A full matrix liquid crystal display graphics panel is incorporated on the RL-H1000. It displays graphics, foreign alphabet characters and proportional spacing, as well as full upper and lowercase ASCII. A ten-speed display control key gives information at varying rates of speed.

The RL-H1000 has a built-in RAM, so you can store up to 500 characters for use as a portable, electronic memo pad. A self-contained world time clock displays month, date, hour and minute. The HHC has a programmable timer with alarm and message function to work as an electronic secretary. It is also compatible with virtually every computer data bank via the telephone modem/acoustic coupler.

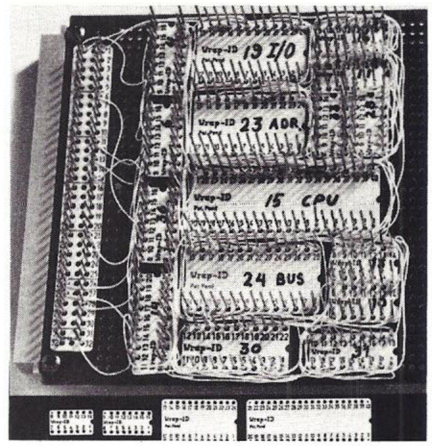
Panasonic Company, One Panasonic Way, Secaucus, NJ 07094. Reader Service number 483.

## Socket-Wrap ID

Now you can easily identify pin numbers on wire-wrapping sockets with the Socket-Wrap ID from OK Machine and Tool Corporation, 3455 Conner St., Bronx, NY 10475. These socket-sized plastic panels with numbered holes in the pin locations can be simply slipped onto the socket before wrapping. You can also write on them for easy identification of location, IC part number or function to simplify both initial wire-wrapping and subsequent troubleshooting or repair. Reader Service number 498.

## Direct-Connect TRS-80 Modem

Lynx is a new direct-connect telephone modem for the TRS-80 that eliminates the need for a separate expansion interface, interface board, telephone coupler and communications



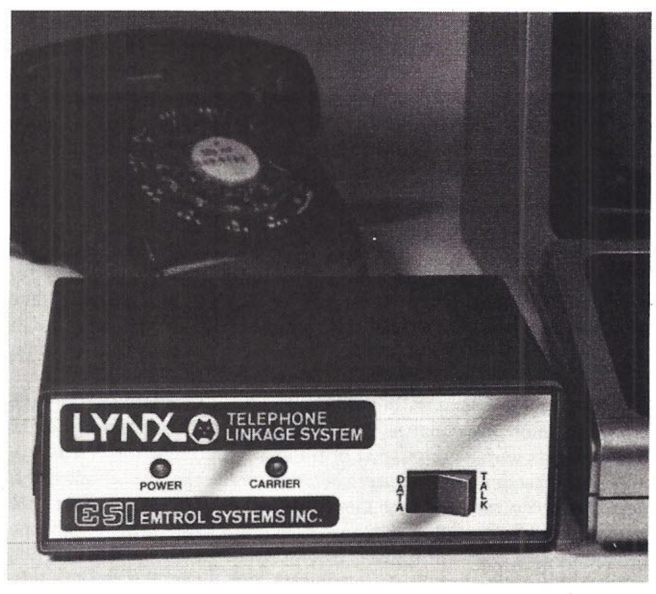

*OK Machine and Tool's pin number identifier.*

software. It connects directly with the keyboard and the telephone line; no acoustic coupler is used. It includes originate and answer capability and is programmable for word length, parity, number of stop bits and full or half duplex. During data exchange, it automatically disconnects the local telephone handset, thus eliminating room noise pickup typical of acoustic couplers.

Minimum hardware requirements are a Level I or II with 4K RAM. Including a terminal cassette program, instruction manual and power pack, the Lynx costs $239.95.

Emtrol Systems, Inc., 1262 Loop Road,

*Panasonic's HCC with peripherals.*




*The Lynx modem.*

*The TRS-80's Color Computer, Model III and Pocket Computer.*

Lancaster, PA 17604. Reader Service number 480.

## Three New Computers From Radio Shack

Radio Shack, 1800 One Tandy Center, Fort Worth, TX 76102, recently unveiled three new TRS-80 computers: Model III, Pocket Computer and Color Computer.

The TRS-80 Model III desktop computer is designed for more data storage, greater versatility and higher computing speed. It is housed in a single cabinet that includes a 65-key keyboard, 12-inch high-resolution video monitor, power supply and space for up to two built-in double-density disk drives. Available in several configurations, it is priced from $699 for the 4K version and is expandable to 32K with 313K of disk storage for $2495. Model III BASIC is compatible with most Model I programs.

The TRS-80 Pocket Computer features a large 24-character LCD display with English language prompting and BASIC programming. It includes 1.9K RAM that retains information for the 300-hour life of its internal batteries. The six-ounce, seven-inch-long pocket computer can be used as a calculator to edit, store, review and place numbers in mathematical equations with up to 15 levels of parentheses. Tapes can be loaded with an optional cassette interface and may also be used to store programs and data. Price is $249.95.

The Color Computer features high-resolution color graphics using any home color TV as a video monitor and instant-load Program Paks. It has a 53-key typewriter-type keyboard, a screen format of 16 lines, 32 characters per line, graphics array from 32 × 64 to 196 × 256, 1500 baud cassette interface and RS-232-type serial interface. In addition to using the plug-in Program Paks, you can program the computer in BASIC and control the color graphics, sound, data manipulation and storage. User programs and data may be stored on an optional cassette recorder. Utilizing its built-in RS-232C serial interface, it can serve as a TRS-80 Videotex terminal with optional software and modem. The 4K RAM, 8K ROM version sells for under $400. Reader Service number 482.

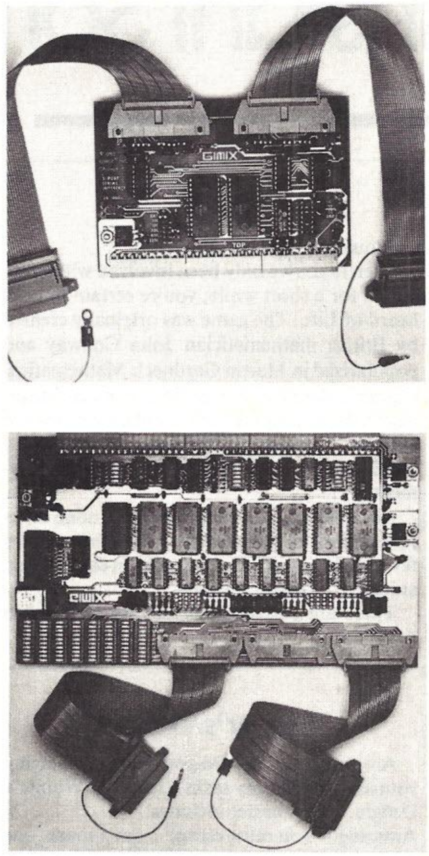
## TRS-80 Lowercase Modification

The TRS-80 lowercase hardware modification from Integrated Service Systems, Inc., 1011 West Broadway, Minneapolis, MN 55411, includes a printed circuit board that connects to the CPU board through a ribbon cable and several wires. You have to remove a 2102 video RAM and install a socket in its place for the ribbon cable. This is not for the novice in soldering iron techniques. It is very easy to lift an etch by applying too much heat. I suggest having a professional do the installation, unless you are experienced with the tools of the trade. The instructions for installation come in a step-by-step format and are easy to follow.

The modification allows 32 additional graphics characters and an indicator in the lowercase mode. Lowercase characters feature pseudo descenders. The modification comes with software drivers on cassette in 4K, 16K, 32K and 48K format. The cassette I received had a bad 16K dump, so I had to use the 4K to look at the operation of the mod. The drivers were single dumped on only one side of the tape. I would like to see at least one set of drivers dumped on each side of the tape to give you a second chance if one of the dumps is bad.

The driver to activate the mod is unique. I tried the mod with several other drivers (one was called a universal driver) without finding any compatibility. You have to use the supplied driver, which will have to be incorporated into any word-processing software you buy for your system. The professionally constructed hardware modification is one of the cleanest I have seen. However, I wish the mod worked with a more universal type of driver. Price is $41.95. Reader Service number 475.

**Edward Umlor**
**Fitzwilliam, NH**



*The Gimix two-port and eight-port serial I/O boards.*

## 6800/6809 I/O Boards

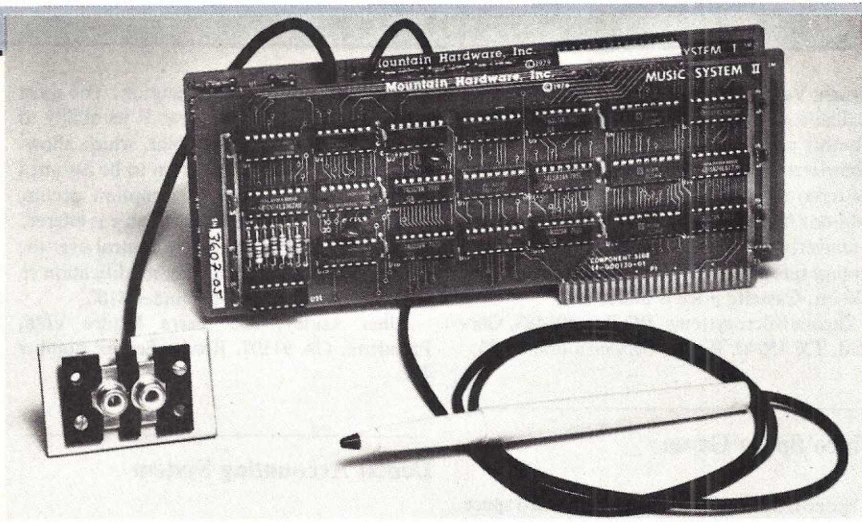
The Gimix 2 Port Serial I/O board has two independent RS-232-compatible I/O ports, with handshaking, on a single 30-pin board. It features jumper-programmable connector pinouts for easy cabling, independent baud rate and interrupt jumpers for each port and the 6850 ACIA. The board is compatible with both the SS-50 (four addresses per I/O slot) and SS-50C (16 addresses per slot) bus configurations. Price is $128.43.

The Gimix 8 Port board features eight independent RS-232-compatible I/O ports on a single 50-pin board. It includes DIP-switch selectable baud rates for each port, extended address decoding for the SS-50C bus, selectable interrupts and the 6850 ACIA. The board is available with an onboard baud rate generator for baud rates up to 38.4K baud. Price is about $300.

Gimix, Inc., 1337 West 37th Place, Chicago, IL 60609. Reader Service number 481.

## Digital Music Synthesizer

MusicSystem is a 16-voice digital synthesizer that permits the creation of the sounds of real musical instruments for the Apple II. The generation of sounds is accomplished through fully programmable waveforms, envelopes and am-




*Mountain Computer's MusicSystem.*

plitudes for each musical "voice."

Provided with the hardware system is software for editing and playing musical compositions. The Editor program permits graphical input of sheet music utilizing standard music notation. The player program permits polyphonic performance of musical compositions. Stereo output is possible through the stereo amplifier and speakers or directly off the card with stereo headphones. Price is $545.

Mountain Computer, Inc. (formerly Mountain Hardware), 300 Harvey West Blvd., Santa Cruz, CA 95060. Reader Service number 477.

## 132/80 Column Printer

The MS-204 is a bidirectional, 9×7 dot matrix printer that accommodates 40, 66, 80 or 132 characters per line. It features a print head life of 100 million characters, a 125 cps print speed and a throughput print speed of 63 lpm. The adjustable sprocket feed mechanism allows use of forms from 2.5 to 9.5 inches wide, with loading from either the bottom or rear. A full 96 ASCII set permits printing upper and lowercase characters that can be expanded for double width fonts in boldface. The VFU (vertical format unit) provides pre-programmed/programmable tab positions, top of form and bottom of form.

The MS-204 is compatible with TRS-80, Apple, PET, Sorcerer or any other Centronics-type system. Price is $795.

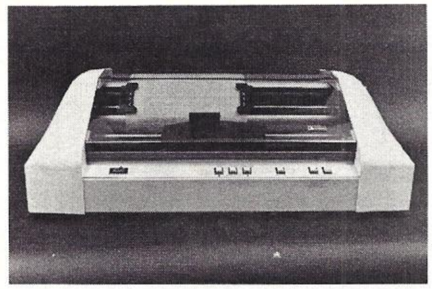
Matchless Systems, Dept. 7, 18444 South Broadway, Gardena, CA 90248. Reader Service number 479.

*Matchless Systems' 132/80 column printer.*

## Anadex Line Printer

The DP-9500 is a dot-matrix line printer with one of the nicest executed designs I have seen. The unit is light and easy to carry from one setup to another. It features built-in parallel, serial and current loop interfaces, as well as two type fonts (9×9 and 7×9) for 10, 12 or 13.3 characters per inch and six or eight lines per inch. Double width format is also available in both fonts. Speed ranges from 150 cps to 200 cps, depending on the type selected. Graphics are individual dot addressable in a 7×1 format. All parameters (except I/O format) are software selectable, as well as switch selectable. The printer will handle edge-punched paper from 1.75 to 16.875 inches.

It takes only a few minutes to unpack and set up this printer. Since it was already selected for parallel (Centronics-format) operation, the printer was hooked up to a TRS-80 word-processing system. The printer performed correctly the first time out. The ready line holds data from going to the printer prematurely, so that all characters were present.



*The DP-9500 Line Printer from Anadex.*

The print quality in both fonts is crisp and clear. A lever, with detent stops, at the left side of the carriage controls the distance of the print head from the platen. Just set the lever for maximum separation to load the paper more easily, and then set it back for the correct impact pressure. This is a much better arrangement than the fixed gap method used by most printers.

The operation was flawless, except for incorrect perforation skip on each new page. This might have been caused by paper drag as the paper came out of its box. When I adjusted the skip setting from one inch to half an inch, there wasn't any paging problem.

I recommend this printer as an excellent addition to any system. It is fast, clean and compatible with any system currently on the market.

Anadex, Inc., 9825 De Soto Ave., Chatsworth, CA 91311. Reader Service number 476.

**Edward Umlor**
**Fitzwilliam, NH**

# NEW SOFTWARE

## Braille Translator

The DS Micro Translator is a microcomputer-based, word-processing and braille-translating system designed for use by schools, universities, agencies and businesses that serve and/or employ blind persons. The system provides for both automatic translation of text to braille and conventional, general office word-processing tasks. The system permits a sighted person with no knowledge of Standard English Braille, which is a complex, semiphonetic code, to enter and edit material from newsletters, memos and class notes to full-length books. Entered text is automatically translated and either formatted by the computer to produce high-quality braille copies of the material or formatted without translation for equivalent print copies. The micro translator renders text into braille at the rate of more than 300 characters per second.

For further information, contact Duxbury Systems, 56 Main St., Maynard, MA, 01754. Reader Service number 489.

## Z-80 Data Base Management System

Target/80 is a data base management system for Z-80 microcomputers from Condor Com-

*APF's Space Destroyer for the Imagination Machine.*

puter Corporation, 3989 Research Park Drive, Ann Arbor, MI 48104. It contains many of the features of Condor's DBM-I and is designed for transaction processing applications. Applications, developed using a nontechnical command language, can be created for personnel, accounting, inventory or other business reporting requirements. The new version uses 19 commands, including relational operations for selecting, sorting, appending or posting data. Target/80 is compatible with most Z-80 microcomputers with at least 48K RAM running under CP/M operating systems. Price is under $700. Reader Service number 494.

### General Ledger System

GL, a general ledger program for the TRSDOS 1.2 on the TRS-80 Model-II, provides immediate financial information for your company by keeping a record of all financial transactions. Features include double entry accounting, ISAM and a full 80-column screen display. It is flexible, interactive, menu-driven and provides automatic integration with A/P, A/R and payroll programs. It requires a 132-column printer, a dual disk system and 64K memory. The $129 price includes a reference manual, an installation guide, 15 programs and sample data files on an eight-inch diskette.

Micro Architect, Inc., 96 Dothan St., Arlington, MA 02174. Reader Service number 488.

### Loan Analysis Software

Personal Loan Analysis is a new software package for the Atari 400 and 800 systems that consists of five menu-selected programs:

Loan Repay—allows the comparison of different interest rates, payment periods and present values.

Number of Time Periods—computes the time required to pay back a loan given a set monthly payment amount and a fixed interest rate.

Present Value—computes the amount of a loan available if the consumer knows his maximum monthly payment.

Amortization—computes all pertinent data for the repayment of a loan.

Add-on/Annual Percentage Rate Conversions —converts the add-on rate to the annual percentage rate or the annual percentage rate to the add-on. Cassette price is under $15.

Zapata Microsystems, PO Box 401483, Garland, TX 75040. Reader Service number 493.

### Video Space Game

Space Destroyer is a new cassette video space game for the Imagination Machine from APF Electronics, Inc., 444 Madison Ave., New York, NY 10022. It tests your commandeering skills to maneuver a squadron of three space destroyers against a continuous wall of phasor-firing aliens. One or two players may enter the battlefield. Points are awarded and displayed on the screen. Sound effects heighten the action of the contest. Price is $19.95. Reader Service number 495.

### CP/M Software Guide

The *CP/M Software Summary Guide*, a list of major software used on most CP/M systems, includes summaries of the CP/M operating system, Microsoft BASIC, CBASIC and the CP/M utilities DESPOOL, MAC and TEX. Commands and utilities are explained briefly with examples. This 60-page booklet organizes features alphabetically, so you can find an explanation quickly rather than page through various function sections. Price is $3.75.

Rainbow Associates, PO Box 35, Glastonbury, CT 06025. Reader Service number 491.

### Basic-Aid for CP/M

Basic-Aid, for programmers who work with Microsoft BASIC, allows one or two key entries of over 40 BASIC statements and commands. It also has nine user-definable buffers for frequently used code (two with 64 characters and seven with 16 characters). It includes a configuration program that allows it to be placed anywhere in memory. The program will run on any 8080 or Z-80 CP/M system and is available on Micropolis five-inch and standard eight-inch disks.

Mendocino Software, PO Box 1564, Willits, CA 95490. Reader Service number 492.

### Star Trac BASIC Monitor

The Star Trac extension to North Star BASIC 5.1 offers the first fully interactive debug monitor for *any* microcomputer BASIC. It allows you to insert a breakpoint in the BASIC program and assume full keyboard control over subsequent execution. The most powerful feature of Star Trac is its ability to assert a conditional breakpoint, which allows control over a BASIC program to be assumed when a specified program symptom occurs, such as when the value of a variable is altered. This monitor allows complete control over the BASIC program without any modification to the program itself. Price is under $100.

Allen Ashley, 395 Sierra Madre Villa, Pasadena, CA 91107. Reader Service number 497.

### Dental Accounting System

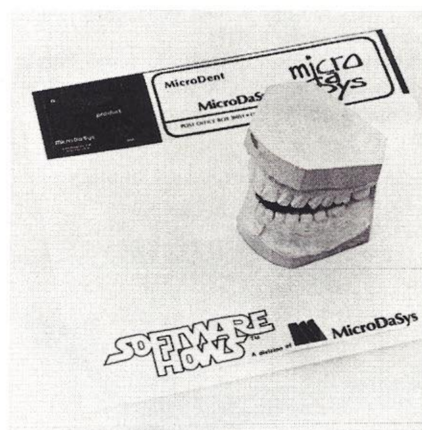
Now your office computer can handle complete dental billing with MicroDent, a dental billing system that will automate your billing procedure, improve turnaround of insurance form processing and provide mail-list and information processing for your patient records. This program keeps track of services rendered, bills patients and/or insurance organizations, handles insurance pre-authorization and prepares statements. It handles insurance form types, or you can easily modify the formats that are used in the system if new forms are added or existing forms are altered.

You can sort through patient information and prepare a mailing list using 15 sort criteria keys. With a form letter merge utility, you can send personalized form letters, dunning messages or notices to selected patients. MicroDent (under $1000) is available in most popular disk formats for CP/M-based systems.

MicroDaSys, PO Box 36275, Los Angeles, CA 90036. Reader Service number 487.

### Apple II File Management System

Filemaster II is a file management system for storing, classifying, manipulating and retrieving data in the Apple II microcomputer. The system includes four Applesoft programs—File Designer, Search and Retrieval, Sort Information and File Converter. Program features include computed numeric fields from user-entered formulas, provisions for creating a sub-




*MicroDaSys' dental billing system.*

file onto a second disk and retrieval of both active and nonactive records, as well as special input routines and error trapping. It requires 48K, Applesoft ROM and a disk drive. Price is under $100.

Rainbow Computing, Inc., Garden Plaza Shopping Center, 9719 Reseda Blvd., Northridge, CA 91324. Reader Service number 496.

## TRS-80 Assembly-Language Development Package

Racet Computes, 702 Palmdale, Orange, CA 92665, has recently released an extended development package for Model II assembly-language programmers. The package includes the following programs:

Macasm—an editor/assembler that includes macro conditional assembly capabilities, in-memory compiles and debug facilities. Source programs can be saved on disk and subsequently reloaded into memory. A range of lines can also be loaded or saved.

Szap—provides the capability to read and modify any sector on a diskette and provides a generalized facility for copying any number of sectors from one area (or disk) to another.

Dis2—a system for the disassembly of Z-80 machine-language code, which can be from memory or from a standard DOS load module from disk. Price is $125. Reader Service number 485.

## ESP Lab

*Reviewed by Eric Maloney,*
Microcomputing *staff*

It was only a matter of time before someone came out with a computer program to test extrasensory perception. But while Manhattan Software, Inc. (PO Box 5200, Grand Central Station, New York, NY 10017), claims that ESP Lab is designed for "serious research," the program will have to undergo some careful scrutiny before it can be accepted as a legitimate tool for scientific study.

ESP researchers have spent years trying to establish the credibility of their work. While they have an endless supply of anecdotal evidence that we humans have extrasensory powers, proving it in the lab has been another story. Trying to isolate and identify ESP to the satisfaction of the scientific community is like trying to catch smoke rings with a butterfly net. A computer only adds another variable to be accounted for.

ESP Lab ($9.95; for the 16K TRS-80, Level II) tests for three types of ESP—telepathy, clairvoyance and precognition—and for telekinesis. Except for the telepathy test, the program can be used by two people—a tester and a respondent—or alone.

The ESP tests are based on the experiments of Dr. J. B. Rhine, whose work at Duke University was instrumental in making ESP the popular subject it is today. Rhine's experiments were done largely with a deck of 25 cards, each marked with a symbol. The deck included five each of squares, circles, triangles, ovals and double-wavy lines.

ESP Lab simply substitutes the computer for the cards. In the telepathy test, the computer randomly selects a symbol, and the keyboard operator tries to mentally project an image of the symbol to another person. In the clairvoyance experiment, the computer selects a symbol and presents a question mark; the tester or respondent enters the guess. In the precognition test, the respondent tries to guess what symbol the computer is going to select. In all three cases, after 25 symbols, the computer displays the guesses and correct answers.

Several problems become immediately evident. To begin with, while the program is based on the Rhine cards, the symbols are actually abstractions generated inside the computer. The respondent thus has no physical object to focus on. This may or may not be important to someone with extrasensory abilities, but has to be a consideration for the serious experimenter.

Also, if the experimenter is self-testing, the tendency is to guess the keys that represent the symbols, rather than the symbols themselves. Again, this may not be critical, but researchers can't dismiss the possibility that it may influence test results.

The telekinesis test is better suited to computer use. A box with a dot in the middle appears on the screen. The user indicates whether he will try to mentally influence the dot to go left or right. Eventually, the dot moves to one side or the other. If the dot moves to the indicated side more than half of the time, this is possible evidence of telekinesis.

No matter what the test, the serious student is faced with major questions concerning the influence the computer has on the results. It is impossible to say whether electrical energy has an effect on telepathic abilities. Some scientists have hypothesized that telepathy might actually be carried on electromagnetic waves. And anyone living near high-tension lines can testify to the deadly impact electricity can have on plant and animal life.

From here, we get into even more esoteric questions. Might some people be more compatible with computers than others, and thus be better receivers? Could some computers be better transmitters than others? Can a computer have ESP?

Most of these questions will concern the serious experimenter only. For others, ESP Lab has a variety of useful applications. For example, it is ideal for the psychology or math student studying probability and statistics, and makes a perfect demonstrator for a class investigating ESP and the mind. In both cases, students can run through the tests quickly and easily, and can get printouts for permanent records or future analysis.

This program is also good for the curious individual who simply wants to make an informal study of the subject. A casual investigation can often be more productive than one conducted in a more formal setting.

Finally, ESP Lab is an excellent party game. It has the appeal of keno or roulette—no matter whether you believe in ESP or chance, there is always the urge to try beating the odds.

By the way, you can test yourself for telekinesis without this program. Using your mental powers, try turning your computer off.

Reader service number 499.

```
CLAIRVOYANCE TRIALS

SUBJECT: ERIC MALONEY
  DATE: 7/14/80

TRIAL # 1

       SYMBOL    RESPONSE

    1  CROSS     TRIANG
    2  LINES     SQUARE
    3  LINES     SQUARE
    4  OVAL      LINES
    5  CROSS     OVAL
    6  CROSS     LINES
    7  SQUARE    TRIANG
    8  TRIANG    CROSS
  * 9  CROSS     CROSS
  * 10 CROSS     CROSS
    11 CROSS     OVAL
    12 TRIANG    LINES
  * 13 SQUARE    SQUARE
  * 14 SQUARE    SQUARE
    15 LINES     TRIANG
    16 TRIANG    LINES
    17 LINES     SQUARE
    18 OVAL      TRIANG
    19 SQUARE    TRIANG
    20 SQUARE    LINES
    21 CROSS     OVAL
  * 22 SQUARE    SQUARE
    23 TRIANG    SQUARE
    24 CROSS     LINES
    25 CROSS     SQUARE
CORRECT RESPONSES:  5
PRECOGNITION SCORE:  7
POSTCOGNITION SCORE:  6
```

*Sample run 1. The program lists the symbols selected by the computer, and the subject's responses. The precognition and postcognition scores indicate the number of times the receiver correctly guesses the previous or next symbol.*

```
TRIAL RESULTS SUMMARY

#   CORR.  PRECOG  POSTCOG

1    5      7       6
2    2      5       3
3    2      6       3
4    9      6       4
5    4      4       8
6    7      4       4
7    3      3       3
8    4      8       6
9    3      4       4
10   6      5       6

AV. CORR. RESPONSES:   4.50
AV. PRECOG SCORE:      5.20
AV. POSTCOG SCORE:     4.70
```

*Sample run 2. Trial results summary. A score of 5 is considered average.*

# LETTERS TO THE EDITOR

## Teachers and Computers

As noted in your June issue of *Kilobaud Microcomputing*, the area of investigating the possibilities and potentials of microcomputers in the classroom is still "virgin," in that the industry hasn't sincerely begun to address the aspect of microcomputers as instructional tools. Granted, there is an abundant source of software for drills and practice. For public school systems, this may be sufficient in itself, if only to relieve the strains of a high student-to-teacher ratio. In our school, with a low (15:1) ratio, such drills are considered superficial to the learning process. A nucleus of parents and faculty has just formed to investigate the possibilities and potentials of microcomputers in the classroom.

While a handful of parents recognize the importance of introducing human-to-processor communication to our children at as early an age as possible, we're constantly confronted with the question of how we should implement the processor into the curriculum as an instructional tool. Specifically, we envision utilizing the computer(s) for an introductory course in programming (primary unit). An area of application, sidestepped too long, is simulation. We would like to see the computer used to reinforce the learning process through automated problem analysis or trend analysis based on variables that are input.

We are now at the point where we (faculty and parents) would like to touch bases with individuals who have successfully gone this route. If those in the field of education or the microcomputer industry can supply us with concrete, logical steps toward implementation of such a program, it would be sincerely appreciated.

**Art Lane**
**78 Whitney Dr.**
**Meriden, CT 06450**

**William Murdoch**
**60 Schooner Lane**
**Meriden, CT 06450**

## Counting 0,1,2...

In "Questions and Answers on Memory Devices" (July 1980, p. 164), the author has done a disservice to novice computerists with his answer, "...just because," referring to a question about track 0.

As a legitimate question, it deserves a similar answer. Zero is the track's identifying number, not a track count. Computerists, like everyone, count the first item as one; however, that item's "name" is often zero for a good reason. In any number system, using 0 to identify an item allows you to distinguish, with a single digit, a number of items equal to the base (ten items for decimal, 16 for hexadecimal, eight for octal). Ignoring 0 requires the use of a second digit, hence, more memory, to identify the same number of items.

Granted, with 77 tracks, this point is obscure; however, the consistent application of this convention leads to some degree of memory conservation.

**P. V. Piescik**
**Wethersfield, CT**

Mr. Piescik's point is well taken. Nonetheless, I don't think many people use the zero numbering convention for the sake of conserving memory. In most cases, either approach will work equally well. (Referencing 77 tracks on a floppy disk, for instance, will require seven bits, regardless of whether the first track is 0 or 1.)

Although my original answer may have seemed flippant, it is essentially correct. The zero numbering convention is simply an established design practice that novices might as well go along with.

**David Price**
**Midlothian, VA**

## A Better Terminal

Frank Derfler's OSI C1P terminal program ("Dial-up Directory," July 1980, p. 68) is most interesting and useful. Regarding the RS-232C information, the signal voltages are allowed to be between $\pm 5$ and $\pm 25$ V dc. Positive voltage is a logic 0 on the data lines and a logic 1 on the control lines. The need for an external TTL-to-RS-232C interface circuit between the OSI output and the modem input is made unnecessary by cutting the trace identified as W10 on the foil side of the board. A negative 5 volts is then connected to J3, pin 7 (and not to the modem). This will allow positive and negative voltage swings as required by the RS-232C definition.

**John G. Ruff**
**Minnetonka, MN**

## SWTP Is No Fun

Why doesn't anyone make any "fun" software for the SWTP 6800? Business programs, operating systems, file sorting and management and letter addressing are available for us home types with small systems, but we're not all serious, you know.

Those TRS-80, Apple and PET guys have all the fun with Chess, Adventures, Flight Simulation and Interlude. There is nothing available for the SWTP in this vein, except some old TSC programs that are quite simplistic by today's standards. In any event, a very finite number of these are extant, with nothing new in sight. Now that our beloved SWTP has all but abandoned us to their fancy 6809 system, we are adrift in space.

Not that I would trade my 6800 for anything on the market. I like to be able to open the cage to view boards that even an idiot would understand. I like the 6800's easy-to-use instruction set. My primary interest is hardware, and the SWTP is great for those of us who want to build special boards and otherwise tinker.

More of us SWTP types should write to thank *Kilobaud Microcomputing* for their support. Listen, SWTPers, if *Microcomputing* stops supporting us because they think we don't care, all we'll have is either a specialty newsletter that thinks that everyone has a disk system with gigabytes of memory and otherwise publishes gossip or all those pretty, colorful, commercial magazines like you-know-who. And what about the Gimix and Smoke Signal types? Are they only interested in file sorting and management? Wouldn't they also like to play games, even just a little?

Lest all the other types smirk, the handwriting is on the wall. TRS-80 Level I is soon going to be in the same boat. Let them try a turn at the bailing bucket.

**John Tavares**
**San Jose, CA**

## Pay Now or Pay Later

John A. Bryant's article and program are well-written ("Calculating Interest Rates," July 1980, p. 134). However, I believe there is a flaw in the program, because it doesn't output the correct interest rates.

If I borrow $1000 from the bank at six percent for one year, the interest is $60. My monthly payment is $1060/12, or $88.33. If I enter this data in the program (principal, $1000; monthly payment, $88.33; and term of one year), the computer says the interest rate is 10.89 percent.

**Ken Walters**
**Red Ash, VA**

It does look like six percent, Ken, and, in fact, when I was a young fellow it was called a six percent loan. The Truth in Lending Act changed that and requires that the annual percentage rate (APR) be given on a loan. In your example, the APR is 10.89 percent, just as the computer said. The loan would have been a six

percent loan if there were no monthly payments and you simply paid back a lump sum of $1060 a year after borrowing $1000. But when you pay it back in monthly installments, you don't have the use of the entire sum for the full year, so the effective interest rate is higher. Thanks for your compliments on the program and article; I probably should have explained APR.

**John A. Bryant**
**Holcomb, NY**

## Shopping Around

I am a professional writer who is keenly and sincerely interested in buying a word-processing system. After reading many books and magazines on the subject to attain a smattering of understanding of the possibilities of what's out there in the market, I stuffed my checkbook into my pocket and eagerly set out to buy. Two months later, after visiting a dozen or so computer stores in Toronto and Buffalo, I still haven't bought a system, simply because nobody will take the trouble to sell me one.

It is my finding that the staff and owners of computer stores come in two types. Type one sneeringly bedazzles the tyro shopper with an incomprehensible jargon of "forty-eight-kay-bus-byte-zed-eighty-menu-interface-vabnagraph!" Type two asks "Whadaya mean, word processing? Wanna see the video games? And it'll do your income tax really neat!"

I respectively ask owners of computer stores to look a little more closely at the way in which their sales staff responds to customers to ensure that their staffs acknowledge that a good deal of would-be buyers drift out the door from lack of adequate service. Finally, ask your sales staff to try a little harder than a perfunctory "You'll need that blue box over there and that black dingus there, and the rest's in this manual." Anybody want to sell me a word-processing computer? Please?

**Sidney Allinson**
**24 Ravencliff Crescent**
**Scarborough, Ontario**
**Canada**

*It's little wonder that some computer stores are having difficulty. How many potential users have been turned off to micros because of treatment like you received? How many sales have been lost?—Editors.*

## Computers on the Air

Aficionados of 6502 microprocessor personal computers have new ways to exchange comments and information. Three new radio nets on amateur radio frequencies have been announced.

An East Coast Apple Net now operates on Saturday mornings at 1300 GMT (9 AM, Eastern Daylight Savings Time) on or near 7260 kHz. Transmission mode for this 40 meter net is lower sideband, with W1UKZ in Scituate, MA, as net control.

In the greater Boston area, there is a new 2 meter net, on the Norwell repeater (144.65/145.25 MHz) for those interested in Apple computers. W1UKZ, WA1ZKB and others act as net control. This net is called to order at 8 PM local time, Wednesday evenings.

A new Atari International Computer Net now meets at 0100 GMT, Tuesdays (9 PM EDST, Monday evenings) on 20 meters. Look for this net on upper sideband around 14.329 kHz. W1UKZ in Scituate, MA, acts as net control.

Aside from conversation about computers in general and the subject computers in particular, these nets will act as funnels for as much official information about computer developments as can be gleaned from the manufacturers. Program swapping via the mails is undertaken, with projected exchange via radio anticipated as more and more computer owners interface with their radio equipment.

**David Allen**
**Scituate, MA**

## Designing Alphabets

Your design for a computer-readable handwritten alphabet ("Publisher's Remarks," August 1980, p. 6) is similar to a project that I have been working on for several years. We spent considerable time and effort developing a new method of handwriting at an early childhood Montessori school (Ashdon Hall in Atlanta, GA). As a side project, using the same pedological techniques, I developed an alphabet design that required a 12-segment LED for electronic display, but readability is easy for the average, untrained person. Readers who are interested in this project can contact me. This method of handwriting for computer recognition could be taught in all elementary schools as part of teaching handwriting.

**Hugh S. Hunt**
**12219 River Road**
**Potomac, MD 20854**

## Identifying PET ROMs

I enjoyed "A 'Personable' Calendar" by G. R. Boynton (August 1980, p. 168). As he said, it is a program just waiting for you to personalize. While some might criticize its lack of some of the programming niceties, I very much enjoyed the program, which did exactly what he set out to do.

My comment does not apply specifically to his program, but to PET programs in general. Since CBM has chosen to have a few different ROM sets, we have to look out for the differences, since not all PET programs will run on all PETs. It would help us true novices (not programmers or electronics types) if the programs were identified for their ROM sets, so we would know whether to convert all POKE/PEEK statements.

**Gerald Key**
**Gahanna, OH**

*Good idea, Gerald. Future PET authors, take note.—Editors.*

# We see the S-100 a little differently.

## We mass-produce S-100 products to deliver industrial quality, at industrial prices.

You systems builders who need top quality, full featured, *workhorse* S-100 building blocks at the most competitive prices now have a source. California Computer Systems.

Industrial quality means top grade materials, components, and assembly, plus complete testing for absolute reliability.

Industrial quality means solid designs, a full complement of the important features you require, and a product line that delivers performance.

Industrial pricing comes from mass production. We buy at the right prices, and build *in quantity,* using state-of-the-art facilities and techniques. Including complete burn-in, for full performance right off the shelf.

Our industrial point of view means you get higher performance, greater reliability, and lower prices. If these are features you would like to see in your S-100 system, see things our way.

Because for serious users with serious uses for the S-100, these are the industrial standards.

**California Computer Systems**
250 Caribbean Sunnyvale, CA 94086 (408) 734-5811

✓148

*James Owens*
*Professor of Management*
*American University School of Business*
*Washington, D.C.*

# Computerized Estate Planning

*This OSI program takes the tedium out of calculating and sorting the various options for the final settlement of your wealth.*

Many people spend a lifetime building an estate (often worth hundreds of thousands of dollars) only to let it go to waste at their death. They plan their lives successfully, but do little or no planning for the disposition of their tangible worth. Much of the estate is thus diverted into estate taxes, probate costs, administrative expenses and a legion of other costs, instead of being channeled efficiently—and mainly intact—to surviving loved ones.

To avoid such problems, the responsible person plans for the inevitable event. He examines his family's future rationally, coolly and in ways that maximize the net estate actually received by intended beneficiaries. The financial community calls this estate planning, and anybody can handle its legion of complexities and mathematical computations with a small computer (and, of course, a lawyer).

Estate planning is a must for all working-age people, not just senior citizens. A 50-year-old man might suffer a sudden heart attack anytime, or a 30-year-old divorced mother might die in an automobile accident. In either case, an unplanned estate usually means hardship for survivors, for whom they worked so hard during life. Given the obvious necessity for priority attention to a personal last will and estate planning, why do so many neglect such a vital matter? Three reasons, mainly: they're

| Enter in line | Variable and sample entry | Description of Variable |
|---|---|---|
| 810 | Y = 1979 | Base year for all data entries and calculations |
| 820 | B = 50 | Wife's age in base year |
| 830 | J = 51 | Husband's age in base year |
| 840 | TI = 20000 | "Death benefit" from husband's retirement program |
| 850 | PL = 2000 | Annual increase in the value of TI (above) |
| 860 | YD = 1999 | Estimated or assumed year of wife's death |
| 870 | A = 150000 | Life insurance (except amount in line 840 above) |
| 880 | SA = 5000 | Total cash, checking accounts and savings in base year |
| 890 | S1 = 1000 | Estimated or assumed future annual savings |
| 900 | HV = 50000 | Value of home in base year |
| 910 | P = .06 | General inflation (cost of living) rate for prices |
| 920 | K = .10 | Average rate of earnings (income) on invested funds |
| 930 | S4 = 500 | Widow's Social Security monthly income in base year (if she were age 60 in base year and awarded maximum) |
| 940 | ET,E = 650,322 | Wife's monthly expenses (if husband dies in base year) in two parts: those subject to inflation; home payment that is not |
| 950 | PR = 40000 | Mortgage (principal) on home in base year |

*Table 1. List of data variables and sample data.*

too busy with the demands of a busy life to think about death; it is most unpleasant, if not traumatic, to confront the issue of one's death and make cool, rational plans for it; and the arithmetic involved in estate planning (calculating the projected income for survivors, varying incomes for varying insurance amounts, effect of estate taxes, etc.) is too time-consuming and complicated. And yet, for many, the most critical act they will perform—or not perform—in life will involve estate planning.

If you have, or have access to, a computer (even a small one limited to just 4K of RAM memory), the program described here can easily remove the major obstacles to your estate planning. It does all the necessary calculations, evaluation of alternative plans and financial projections, quickly enough for the busiest executive or careerist.

### Program Objectives

The program is designed to project anyone's net estate after taxes (as actually inherited by survivors), as well as survivors' monthly income, including Social Security benefits, for any assumed year of death of that person and for any number of years following death. Yearly calculations include changing home value, equity in home, savings and net worth.

The program can also quickly reveal the minimum life insurance required for a particular person and situation and the ultimate effect of inadequate or no life insurance.

Finally, the program might prompt some readers to search out and organize vital statistics about their finances and the data necessary if they are to be in control of their lives.

### Data Required

I have kept data requirements to run the program at a minimum, and they should be easily available to the user. (I permitted the program itself considerable complexity to keep data entry easy and convenient.) Sixteen data are all that need be entered into the data lines (Table 1), and most of these will be immediately accessible. You might have to examine your retirement program to determine its "death benefit" and annual increase in value (for lines 840 and 850); contact the Social Security Administration for the current (base year) amount for line 930 and determine current monthly expenses to realistically project expenses for line 940.

All other data are obvious (such as your age) or assumed (such as projected inflation rate in line 910). Enter these 16 data to run the program. (Table 1 includes sample data for a hypothetical, but realistic, case; program output for this case is displayed in Table 2.) Some of your data might be zero, as in lines 900 and 950, if you are renting rather than buying a residence; in many cases, line 840 and 850 will also be zero.

The sample data used here assume the following: a husband (age 51) and wife (age 50), with no dependent children, owning a $50,000 home with a $40,000 mortgage at nine percent for 30 years and a $322 monthly payment (excluding taxes and insurance), husband's death benefit value and life insurance total of $20,000 plus $150,000, savings of $5000 and an estimate of six percent for inflation in future years.

### Program Output

Program output (Table 2) generates two loops: the first (or outer) loop, always starting with the expression "if husband dies in," displays wife's net cash received (after taxes) and her monthly income including any Social Security benefits for that particular year of his death from 1980 to 1999. The second (or inner) loop, always starting with the expression "at wife's age," displays the wife's monthly income, expenses, monthly surplus (positive or negative), home value, equity in home and cumulative surplus for each year of her life after his death in the year stipulated in the particular outer loop year. (1999 above can easily be changed in line 860 for more or fewer years' projections.)

If this looks complicated, the potential reality it represents is just that—complicated. The wife's income and other variables change depending upon two factors: the year of the husband's death and (for each year of his death) the number of years (her age) following his death.

For example, if he dies in 1985, the base amount of insurance, savings and so on determining the next ten or 20 years of her life's monthly income is far different than the base amount if he died in 1980. In the sample data and output, the program generates 361 inner loop displays (19 inner loop financial situations times 19 outer loop possibilities of death), and all are different depending on the two factors. Fortunately, the program incorporates and handles these complexities, and the program user need only enter base year data (as above) and read the output information for each year.

To illustrate several immediate uses of the program output (still using Table 1 sample data), suppose that the husband dies in 1980. In that year, when the wife is 51, her monthly surplus or margin is a comfortable $431. But by the time she is 59, just before Social Security benefits are activated at 60, her monthly surplus has declined to zero. The reason is that her income is frozen at the level of the year of the husband's death while inflation drives up monthly expenses each year.

Fortunately, in this case, her cumulative surplus from prior years is sufficient to absorb the erosion of the monthly surplus. By contrast, if life insurance had been $100,000 instead of $150,000, the wife's monthly surplus at 59 would be −$419 and cumulative surplus −$20,000 (obviously, the economic system would not extend her such borrowed funds, and the only option would be a drastic collapse in her standard of living).

For anyone using this program, many assumptions can quickly be entered into data lines and run to test their long-term consequences, especially assumptions about minimum life insurance needed at any given time.

### Program Mechanics and Documentation

The program has two loops—an outer GOTO loop and an inner FOR-NEXT loop. The inner loop also incorporates a number of subroutine loops. Many secondary variables (such as BB) and temporary variables (such as V1) are used to isolate the intended effect on one loop from the unintended ef-

fect upon the other as numerical values need programmed changes in both—but at different program points. Using the basic variables in the Table 1 data list as a starting point, it will be easy to trace the logic and movement of the program.

Program documentation (REM lines) has been eliminated and placed instead in the text here to minimize memory required to run the program. As listed, the program can output 20 or more outer loops, each with 20 or more inner loops, and do it within the limits of a 4K RAM memory system. Thus, anyone, even with his first personal computer, can use the program immediately.

Only a few other program lines need special comment:

- Social Security benefits are initiated and "indexed" in lines 116, 131 and 930.
- Estate taxes are calculated in line 126.
- Death benefit (line 840) is frozen at husband's retirement age in line 121.
- Monthly expenses subject to inflation are increased by P factor in lines 133 and 177.
- Calculations for home value, amortization of mortgage and home equity are in lines 310 through 315.

Note that, in line with past years' patterns, the annual increases in home value are set two percentage points above the rate of general inflation (P), and that home value and equity rates of increase decline slightly in the later years of a run to reflect some real depreciation with age. For brevity, I used different formulas to calculate home value and equity in the outer as contrasted with the inner loops; this results in slight, but insignificant, absolute value variations in the later years for each loop during a run.

### Program Modifications

My primary criteria in the program design were convenience and ease of use. At times, however, you will want to modify the program itself. Mainly, there are three such occasions.

The first concerns the estate tax formula built into the program. Current law allows a $250,000 marital deduction when a husband passes his estate, at death, to his wife (provided he explicitly declares in his will his intent to do so—neglect in making a will with such an explicit statement in it disallows the deduction and deprives the wife of about $80,000 to taxes!).

Current law has also set the unified credit against the actual net estate tax at $47,000, deductible from the tax itself. Thus, if you plan your estate properly, you can take advantage of the deduction and credit and pass on an estate of up to $400,000 with no estate taxes at all. But the law changes often—usually every two years with congressional elections. When it does, make the change in the program in line 126 by substi-

```
10 REM        ****************************************
20 REM        ** E S T A T E   P L A N N I N G    **
30 REM        **                                  **
40 REM        **     PROGRAM BY J. OWENS, PH. D. , 1979  **
50 REM        ****************************************
60 DEF FNT(X) = INT(X * 1 +.5)/1
70 DEF FNH(X) = INT(X * 100 +.5)/100
80 PRINT"**************************************************"
100 READ Y,B,J,TI,PL,YD,A,SA,S1,HV,P,K,S4,ET,E,PR
110 YY=Y : BB=B : T=TI : HW=HV : RP=PR : AA=A
115 IF B>59 THEN GOTO 120
116 S4=FNT(S4+(S4 * P * (60-B)))
120 PRINT"IF HUSBAND DIES IN";Y+1;"WIFE WILL INHERIT CASH:"
121 Y=Y+1 : B=B+1 : J=J+1 : IF J<66 THEN TI=TI+PL
123 SA=SA+S1 : TT=TI+A+SA : T9=TT+EH : X=X+1
125 PRINT"INSURANCE";A;"+ DEATH BENEFIT";TI;"+ SAVINGS";SA;"=";TT
126 IF T9>426000 THEN TX=FNT((T9-426000)*.33) : IFT9<426000THENTX=0
128 V=TT-TX
129 IF B < 60 THEN SS=0
130 IF B>59 THEN SS=FNT(S4)
131 IF B>59 THEN S4=S4+(S4*P)
132 VV=FNT((V*K)/12)
133 ET=ET+(ET*P) : EX=FNT(E+ET)
135 PRINT"NET CASH INHERITED LESS TAXES(";TX;") =";  V
136 PRINTTAB(15);"MONTHLY INCOME=";VV;"+SOC. SEC. ";SS;"=";VV+SS
140 FOR R= 1 TO (YD-(Y+1))
145 PRINT
150 PRINT"AT WIFE'S AGE"; BB+1; "(IN";YY+1;")"
160 BB=BB+1 : YY=YY+1 : IF BB>60 THEN GOTO 170
165 IF BB>59 THEN SS=S4
170 IN=VV+SS:SS=FNT(SS+(SS*P)):EE=E7+E8:IF EE>EX THEN GOTO 177
175 E7=E : E8=ET
177 EE=FNT(E7+E8) : E8=E8+(E8*P)
200 EE=FNT(E7+E8) : SU=FNT(IN-EE)
300 PRINT"HER MONTHLY INCOME=";IN; "AND EXPENSE=";EE; "(SURPLUS=";SU
305 IF R=1 THEN HE=HV-PR
310 HV=HV+(HV*(P+.02)) : V1=PR*.09 : V2=E*12 : PR=PR-(V2-V1)
315 HE=HV-PR: REM AMORTIZING AT 9% AND BY YEAR, NOT MONTH
325 PRINTTAB(8);"HOME VALUE=";FNT(HV),"HOME EQUITY=";FNT(HE)
330 CS=CS+(SU*12)
350 PRINTTAB(8);"WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS=";CS
500 NEXT R
510 EE=EX : E7=E : E8=ET
520 BB=B : YY=Y : T=TI : CS=0
522 HV=HW+(HW * ((P+.02)* X))
530 PR=RP-(((E*12)*X)-((RP*.09)*X))
540 PRINT"**************************************************"
600 IF Y=YD THEN END
650 PRINT
700 GOTO 120
810 DATA 1979
820 DATA 50
830 DATA 51
840 DATA 20000
850 DATA 2000
860 DATA 1999
870 DATA 150000
880 DATA 5000
890 DATA 1000
900 DATA 50000
910 DATA .06
920 DATA .10
930 DATA 500
940 DATA 650,322
950 DATA 40000
```

*Program listing. Estate Planning program in OSI BASIC.*

tuting new numbers for the present line's .33 and 426000.

The second possible modification involves life insurance provisions throughout the program that cannot be handled in the data lines. The program assumes a set amount of life insurance (as entered in line 870) until the husband's death. If a portion is term or group insurance, ending at the husband's retirement age of 65, that amount must be deducted by adding a new program line. If, for example, the term amount ending at 65 were $20,000, then add:

Line 122 IF J = 65 THEN LET A = A – 20000

If the amount affected here were $30,000 terminating at age 62, line 122 would become:

Table 2. Partial program output (using data in Table 1).

```
***************************************************
IF HUSBAND DIES IN 1980 WIFE WILL INHERIT CASH:
INSURANCE 150000 + DEATH BENEFIT 22000 + SAVINGS 6000 = 178000
NET CASH INHERITED LESS TAXES( 0 ) = 178000
                   MONTHLY INCOME= 1483 +SOC.SEC. 0 = 1483

AT WIFE'S AGE 51 (IN 1980 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1052 (SURPLUS= 431
          HOME VALUE= 54000   HOME EQUITY= 14264
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 5172

AT WIFE'S AGE 52 (IN 1981 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1096 (SURPLUS= 387
          HOME VALUE= 58320   HOME EQUITY= 18872
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 9816

AT WIFE'S AGE 53 (IN 1982 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1143 (SURPLUS= 340
          HOME VALUE= 62986   HOME EQUITY= 23851
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 13896

AT WIFE'S AGE 54 (IN 1983 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1192 (SURPLUS= 291
          HOME VALUE= 68024   HOME EQUITY= 29232
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 17388

AT WIFE'S AGE 55 (IN 1984 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1244 (SURPLUS= 239
          HOME VALUE= 73466   HOME EQUITY= 35046
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 20256

AT WIFE'S AGE 56 (IN 1985 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1299 (SURPLUS= 184
          HOME VALUE= 79344   HOME EQUITY= 41330
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 22464

AT WIFE'S AGE 57 (IN 1986 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1358 (SURPLUS= 125
          HOME VALUE= 85691   HOME EQUITY= 48120
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 23964

AT WIFE'S AGE 58 (IN 1987 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1420 (SURPLUS= 63
          HOME VALUE= 92547   HOME EQUITY= 55458
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 24720

AT WIFE'S AGE 59 (IN 1988 )
HER MONTHLY INCOME= 1483 AND EXPENSE= 1486 (SURPLUS=-3
          HOME VALUE= 99950   HOME EQUITY= 63388
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 24684

AT WIFE'S AGE 60 (IN 1989 )
HER MONTHLY INCOME= 2283 AND EXPENSE= 1556 (SURPLUS= 727
          HOME VALUE= 107946  HOME EQUITY= 71957
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 33408

AT WIFE'S AGE 61 (IN 1990 )
HER MONTHLY INCOME= 2331 AND EXPENSE= 1630 (SURPLUS= 701
          HOME VALUE= 116582  HOME EQUITY= 81218
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 41820

AT WIFE'S AGE 62 (IN 1991 )
HER MONTHLY INCOME= 2382 AND EXPENSE= 1708 (SURPLUS= 674
          HOME VALUE= 125909  HOME EQUITY= 91226
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 49908

AT WIFE'S AGE 63 (IN 1992 )
HER MONTHLY INCOME= 2436 AND EXPENSE= 1792 (SURPLUS= 644
          HOME VALUE= 135981  HOME EQUITY= 102041
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 57636

AT WIFE'S AGE 64 (IN 1993 )
HER MONTHLY INCOME= 2493 AND EXPENSE= 1880 (SURPLUS= 613
          HOME VALUE= 146860  HOME EQUITY= 113729
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 64992

AT WIFE'S AGE 65 (IN 1994 )
HER MONTHLY INCOME= 2554 AND EXPENSE= 1973 (SURPLUS= 581
          HOME VALUE= 158608  HOME EQUITY= 126360
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 71964
```

```
AT WIFE'S AGE 66 (IN 1995 )
HER MONTHLY INCOME= 2618 AND EXPENSE= 2072 (SURPLUS= 546
          HOME VALUE= 171297  HOME EQUITY= 140010
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 78516

AT WIFE'S AGE 67 (IN 1996 )
HER MONTHLY INCOME= 2686 AND EXPENSE= 2177 (SURPLUS= 509
          HOME VALUE= 185001  HOME EQUITY= 154762
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 84624

AT WIFE'S AGE 68 (IN 1997 )
HER MONTHLY INCOME= 2758 AND EXPENSE= 2289 (SURPLUS= 469
          HOME VALUE= 199801  HOME EQUITY= 170785
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 90252
***************************************************
                    •
                    •
                    •
***************************************************

IF HUSBAND DIES IN 1995 WIFE WILL INHERIT CASH:
INSURANCE 150000 + DEATH BENEFIT 48000 + SAVINGS 21000 = 219000
NET CASH INHERITED LESS TAXES( 0 ) = 219000
                   MONTHLY INCOME= 1825 +SOC.SEC. 1135 = 2960

AT WIFE'S AGE 66 (IN 1995 )
HER MONTHLY INCOME= 2960 AND EXPENSE= 2072 (SURPLUS= 888
          HOME VALUE= 118800  HOME EQUITY= 83380
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 10656

AT WIFE'S AGE 67 (IN 1996 )
HER MONTHLY INCOME= 3028 AND EXPENSE= 2177 (SURPLUS= 851
          HOME VALUE= 128304  HOME EQUITY= 93561
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 20868

AT WIFE'S AGE 68 (IN 1997 )
HER MONTHLY INCOME= 3100 AND EXPENSE= 2289 (SURPLUS= 811
          HOME VALUE= 138568  HOME EQUITY= 104562
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 30600
***************************************************

IF HUSBAND DIES IN 1996 WIFE WILL INHERIT CASH:
INSURANCE 150000 + DEATH BENEFIT 48000 + SAVINGS 22000 = 220000
NET CASH INHERITED LESS TAXES( 0 ) = 220000
                   MONTHLY INCOME= 1833 +SOC.SEC. 1203 = 3036

AT WIFE'S AGE 67 (IN 1996 )
HER MONTHLY INCOME= 3036 AND EXPENSE= 2177 (SURPLUS= 859
          HOME VALUE= 123120  HOME EQUITY= 87988
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 10308

AT WIFE'S AGE 68 (IN 1997 )
HER MONTHLY INCOME= 3108 AND EXPENSE= 2289 (SURPLUS= 819
          HOME VALUE= 132970  HOME EQUITY= 98540
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 20136
***************************************************

IF HUSBAND DIES IN 1997 WIFE WILL INHERIT CASH:
INSURANCE 150000 + DEATH BENEFIT 48000 + SAVINGS 23000 = 221000
NET CASH INHERITED LESS TAXES( 0 ) = 221000
                   MONTHLY INCOME= 1842 +SOC.SEC. 1275 = 3117

AT WIFE'S AGE 68 (IN 1997 )
HER MONTHLY INCOME= 3117 AND EXPENSE= 2289 (SURPLUS= 828
          HOME VALUE= 127440  HOME EQUITY= 92596
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 9936
***************************************************

IF HUSBAND DIES IN 1998 WIFE WILL INHERIT CASH:
INSURANCE 150000 + DEATH BENEFIT 48000 + SAVINGS 24000 = 222000
NET CASH INHERITED LESS TAXES( 0 ) = 222000
                   MONTHLY INCOME= 1850 +SOC.SEC. 1352 = 3202

AT WIFE'S AGE 69 (IN 1998 )
HER MONTHLY INCOME= 3202 AND EXPENSE= 2407 (SURPLUS= 795
          HOME VALUE= 131760  HOME EQUITY= 97204
          WIFE'S CUMULATIVE (MONTH-BY-MONTH)SURPLUS= 9540
***************************************************
```

IF J = 62 THEN LET A = A − 30000

The third possible modification makes the program more convenient to use. As programmed, any output will display in sequence many inner loop years for each outer loop year of the husband's death. For a quicker summary, outputting outer year results only, change line 140 to: FOR R = 1 TO 1.

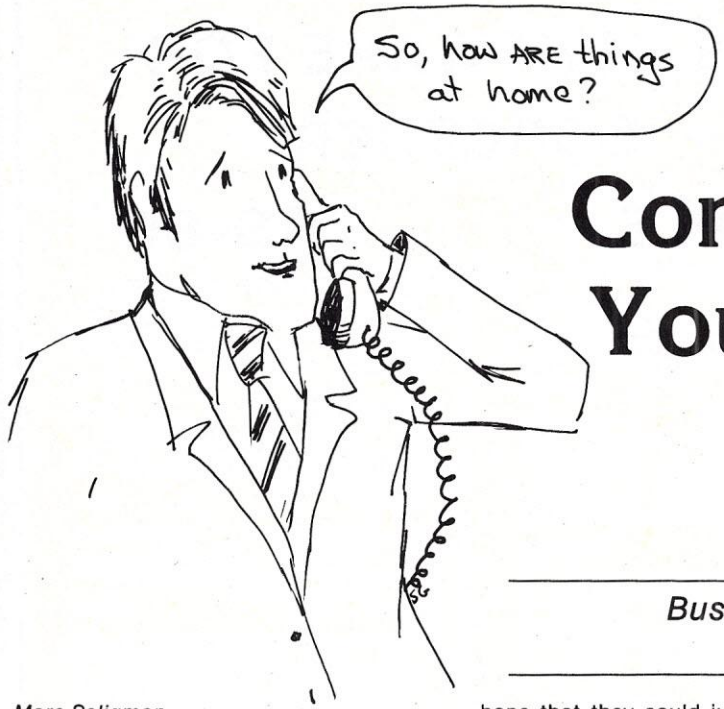
If you want only, say, three inner loop years for each outer loop year, then use line 140 in the form:

FOR R = 1 TO 3

### Conclusion

This program is no substitute for your lawyer's and banker's professional advice about your will and strategies available in estate planning. But it makes the calculations you need to see future objective results of current decisions or options, and helps evaluate those decisions and options. ■


*Dr. James Owens is the director of the Department of Management Sciences and a professor of management and organizational behavior at the American University School of Business Administration in Washington, D.C.*




# Conversing with Your Computer

*Businessmen, call your computer from the nearest phone.*

*Marc Seligman*
*320 Bryn Mawr Ave.*
*Bala Cynwyd, PA 19004*

Remote use of your computer is neither as expensive nor as complicated as you might believe. Follow these instructions and you will be able to tie into your computer from a friend's house, a business meeting or your favorite watering hole.

This article is most useful to owners of the Vector MZ system with the Mindless Terminal. It can also be adapted for other MZ systems and for computers from other manufacturers. You must, however, design your own conversions.

The Vector MZ has enjoyed remarkable success throughout the country, but especially on the West Coast. Due to its popularity as a small-business system, major modifications are often needed. The trouble is, most purchasers have no knowledge of machine coding or assembly language, and don't wish to learn. They want a fast, reliable system. They are not hobbyists, but dedicated users who need their computers for serious business applications. When they have a problem, they want it solved now, without a lot of foot-shuffling and programs-take-time-to-write excuses.

Were these users to ask about telephone interfacing at their computer store, they would most likely be steered to an expensive and complicated internal auto-answer modem. It they were foolish enough to buy an auto-answer external modem with the hope that they could just plug it in, they would be in for a rude awakening.

### Telephone Connections

This method is inexpensive and amazingly effective. The necessary parts include a Vector MZ with Mindless Terminal, a Texas Instruments Model 745 portable data terminal set to mark parity ($1595), a Data Access Systems Model DASI 68-01 modem ($300) and an RS-232 cable (use pins 1, 2, 3, 4, 6, 7, 8 and 20; wire pin 2 of one plug to pin 3 of the other and vice versa). (See Fig. 1.)

If you don't know anything about wiring a cable, have the store employees wire it for you. All wiring is direct except for the pin 2 and 3 exchange noted above.

Before you make the big test, be sure that the printer routine on your RES module is the DECW4 version. Section 2.2.1.6 of your Vector Graphic manual tells you how. If you normally use another version, don't panic. Copy your master diskette, then modify the RES module and save it according to section 2.2.1.6. Just copy any of your programs onto this disk and make sure you use only this disk when you access the computer from your remote terminal.

You also have to adjust the baud rate of your computer to match the speed of the 745 (300 baud). Even if you've never opened up your computer, this is one time that you should. As with all electrical appliances, make sure it's unplugged.

Unscrew the four Phillips-head screws (two on each side) on the top part of the case and lift off the top. On one of the printed circuit boards near the rear of the computer, a series of small switches in the upper left corner will usually be set to 1200. Using a pencil or small screwdriver, pop the switch that is out of line so that it is in line with the others. Next, find the switch labeled 300 and pop that out of line.

When you first open the computer, touch your hand to the metal frame on the left side that acts as a guide for the circuit boards, but don't touch anything else. Some electrical wizards feel that this will ground any loose static electricity in your body and prevent the computer from having a coronary if your hand happens to slip later on in the procedure.

Plug the modem into the wall outlet. Connect its telephone jack to the modular outlet that you already have on the wall. If you don't have a modular jack, Radio Shack sells adapters that you can plug in or screw on to your telephone connector box. Again, if you are mechanically inept, have the phone company install the adapter.

If you have an extra modular cable, you can plug your telephone into the extra modular socket on the back of the DASI 68-01 (a nice feature that saves having to remember to replug the telephone into the wall when you are done with your remote work). Finally, connect the RS-232 cable between the modem and the computer (either side can plug into either device).

### Software

Turn on the computer and bring up BASIC. Enter the program listing and save it. I call this program Remote. The next to last statement loads my menu so that I am

ready to run as soon as the ready prompt is flashed on the Mindless Terminal. For your own use, enter the name of your menu program. If you do not have a menu program, enter the name of any other program you have on the diskette.

Run the program. The computer will set itself for input through the RS-232 connector rather than through the Mindless Terminal. All operations can be performed by direct wiring to the connector (as with a Texas Instruments Model 820 KSR printer) or remotely through a modem connected to the computer.

Press the power button on the modem. A red light indicates that all is fine. Go to another phone and dial the first phone number. A ring or two will be followed by a high-pitched sound. Following the procedures that came with the TI 745 instruction book, put the phone handset into the acoustic cuffs of the 745, which is already turned on, according to the manual's instructions.

Once the connection is made, press the on-line key and make sure that the terminal is set for full duplex and high speed (read the manual for details). Type a control-C to set a little interrupt; the computer will follow on the next line with a ready. If nothing happens, make sure that the green carrier detect light is on, and that the terminal is on-line. The switches for full duplex and high speed will not prevent data from being transmitted, even if they are set incorrectly. But then again, the data will be so garbled that the messages will be meaningless.

### Operation

From your portable terminal you can do whatever you like. The escape key on the portable terminal will not function. A control-X will bring back the BASIC sign-on message but will preserve whatever program you had in memory. A rubout will generate a back space and the ability to change previously entered information. A control-U will erase an entire line's input. Control-S will stop the computer from sending more information. Hitting any other key will resume transmission with no data loss.

With this particular arrangement, you will receive on the 745 an echo of what the computer receives. Local printing is suppressed, so typing will be a little strange, since there will be a brief delay between typing a character and having it appear on the thermal paper.

To reuse your computer with the Mindless Terminal, hit the computer reset button and type in a B from the keyboard on the Mindless Terminal. The entire system will reboot and allow normal operation of the system.

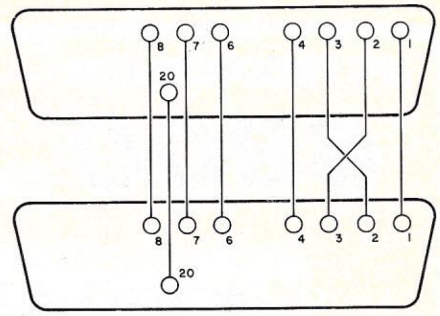
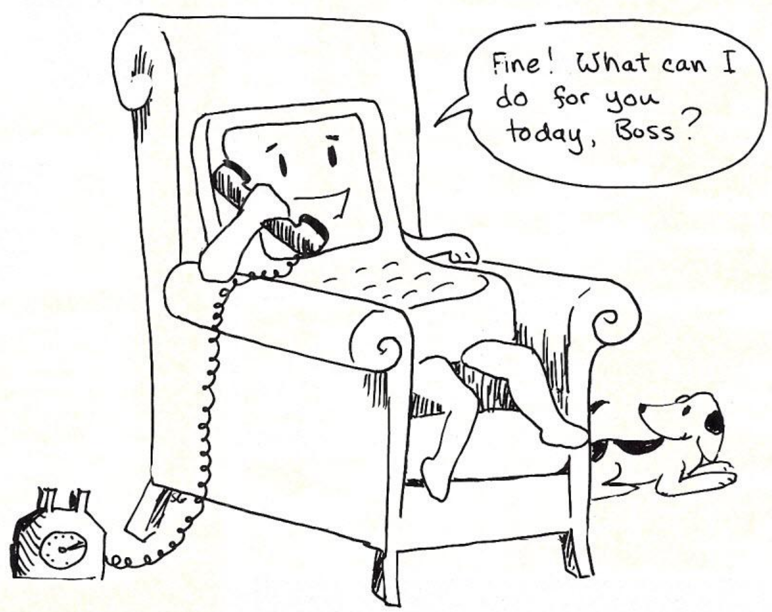



*Fig. 1. Pin assignments for RS-232-C cable.*

To use the computer from a remote location, make sure that the speed is set to 300 baud and execute PLOADG Remote. Test the system with the portable terminal before you leave the computer far behind. If you accidentally load a program other than Remote, the portable terminal will not work, and if you need to access your computer, you won't be able to do so. Also, turn off the modem when not in use, or else people will never be able to reach you by telephone.

The modem disconnects the computer and hangs up the phone after you terminate remote communication. You can call it again and again without having to reset anything on the computer. It is truly an efficient, low-cost method of accessing your computer from the field. ■

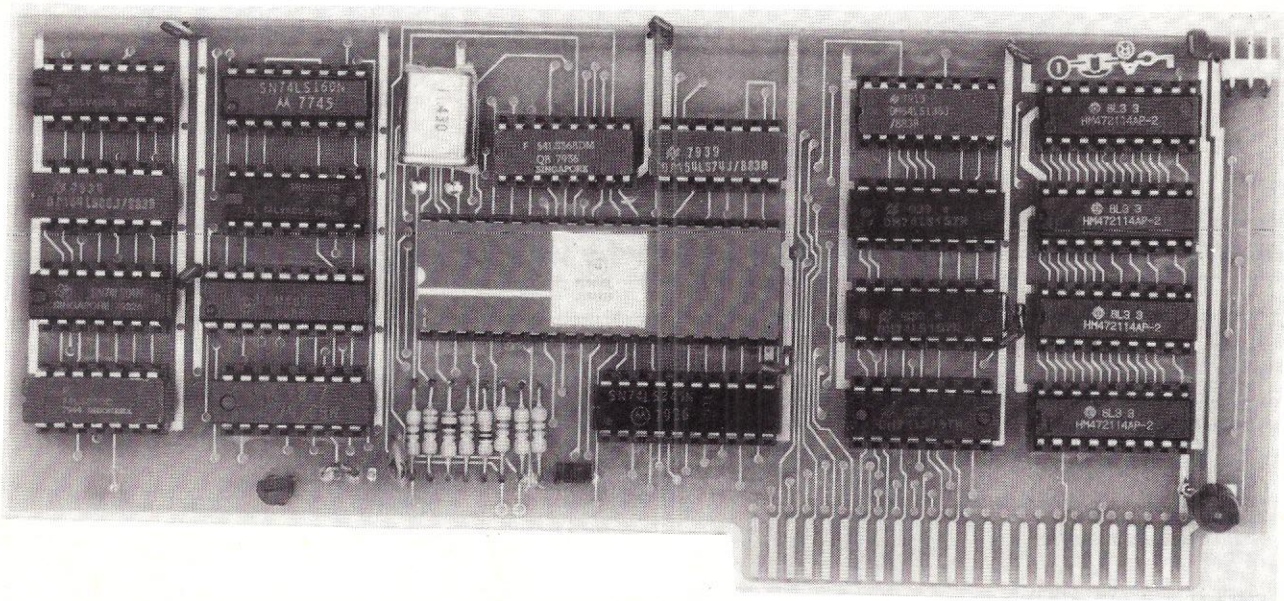
```
10  REM ** REMOTE
20  REM **
30  REM ** THIS PROGRAM SETS UP THE VECTOR MZ SYSTEM B FOR REMOTE USE THROUGH
40  REM ** A MODEM (DASI 68-01) ATTACHED TO ITS RS 232 CONNECTOR.  THE REMOTE
50  REM ** DEVICE IS A TEXAS INSTRUMENTS 745 PORTABLE DATA TERMINAL SET FOR
60  REM ** MARK PARITY.   BAUD RATE ON THE COMPUTER MUST BE SET TO 300.
70  REM
80  REM ** WRITTEN BY:     MARC SELIGMAN
90  REM **                 SEPTEMBER 11, 1979
100 REM **
110 REM ** FOR PERSONAL USE ONLY.  NOT TO BE RESOLD.
120 REM
130 REM
140 REM
150 ASSIGN (2,3)
160 POKE(16R0712)=16RDB
170 POKE(16R0713)=16R07
180 POKE(16R0714)=16REE
190 POKE(16R0715)=16R04
200 POKE(16R0716)=16RE6
210 POKE(16R0717)=16R02
220 POKE(16R0718)=16RC8
230 POKE(16R0719)=16RDB
240 POKE(16R071A)=16R06
250 POKE(16R071B)=16RE6
260 POKE(16R071C)=16R7F
270 POKE(16R071D)=16RFE
280 POKE(16R071E)=16R18
290 POKE(16R071F)=16RCA
300 POKE(16R0720)=16R52
310 POKE(16R0721)=16RC1
320 POKE(16R0722)=16RC9
330 POKE(16R05F9)=16R12
340 POKE(16R05FA)=16R07
350 POKE(16R0605)=16R12
360 POKE(16R0606)=16R07
370 LOAD "MENU"
380 END
```




*Program listing.*

FOR APPLE II AND APPLE II PLUS COMPUTERS

# DoubleVision™



## 80 x 24 Video Display with Upper and Lower Case

COLUMNS LINES

• is a hardware board that may be plugged into any slot in Apple II or Apple II Plus 32K or 48K Disks • full 128 ASCII character set, including control characters • fully programmable cursor • built in light pen capability • inverse video • full cursor control • works with 50/60Hz • has 2k of its own screen memory • has its own video output jack that must be connected to a monitor (or a high band width black & white TV thru a good RF modulator). Color TV's produce a poor display and are not recommended. • permits you to connect another monitor (or a T.V. set thru RFmod) to the Apple video output jack • displays 24 lines of 80 column text — programmable for different values • permits you to have graphics on Apple video output • video output and Apple video output may be connected to one monitor thru optional video switch • is active only when addressed for reading from or writing to • accepts lower case input from keyboard by use of escape key. (no modification required) or direct use of shift key (1-wire connection from shift key pad to DoubleVision required). • is compatible with the latest version of various word processing software packages. Presently these include Apple-pie 2.0— Programma International, Easywriter Professional system—Informational Unlimited, Text Editor/Formatter—Peripheral's Unltd. (when ordering from these companies, please ask for versions compatible with DoubleVision). All software available from Computer Stop when released. • Peripheral's Unltd. B.I.T.S. and P.I.T.S. and Southeastern Software's "DATA CAPTURE" with Micromodem and communication card. These packages give ability to upload, transfer and download files from remote computers, and all at 80 columns! • Programma Int. latest assembler LISA V:20 will support full 80 column display • is transparent for use with Basic and Pascal • software on disk for easy modification and adaptation for different applications • completely commented source listing of software and hardware schematics available • PASCAL (optional) • becomes the console when installed in Pascal • Permits 80 column text processing with full upper/lower case while using Pascal's editor • must be plugged into slot 3 when operating with Pascal

**Available now at your local computer store. . . . . . . $295.00**

Call Computer Stop for Store nearest you

Shipping, Insurance, Handling, extra

Calif. Residents add 6% Sales Tax

*Apple is a Registered TM of Apple Computers, Inc.

*Dealer inquiries invited.*
*Contact:*

COMPUTER STOP CORP.
2545 West 237th St.
Suite L
Torrance, CA 90505
539-7671

The Computer Stop
16919 Hawthorne Blvd.
Lawndale, CA 90260 ✓283
(213) 371-4010

**MON. - SAT.**
**10-6**

# Address List Program

*Machine-language program for 6800 users who want to keep up to date on who's who.*

```
NAME (20 CHARACTERS)
JACKSON; JOHN,
TELEPHONE (11 DIGIT MAX)
3012516486
BIRTH DATE (YYMMDD)
290403
STREET (21 CHARACTERS)
243 AMOS ST.,
CITY;STATE (17 CHARACTERS)
NEWBURG; MO,
ZIP CODE (5 DIGITS)
33557

NAME (20 CHARACTERS)
DAWSON; HAROLD *,
TELEPHONE (11 DIGIT MAX)
4532875496
BIRTH DATE (YYMMDD)
310427
STREET (21 CHARACTERS)
4730 FULTON RD.,
CITY;STATE (17 CHARACTERS)
JACKSON; MI,
ZIP CODE (5 DIGITS)
76321

NAME (20 CHARACTERS)
YARREL; PHYLLIS,
TELEPHONE (11 DIGIT MAX)
4749763287
BIRTH DATE (YYMMDD)
340823
STREET (21 CHARACTERS)
455 53RD ST.,
CITY;STATE (17 CHARACTERS)
SAN FRANCISCO; CA
ZIP CODE (5 DIGITS)
34567

NAME (20 CHARACTERS)
!
```

*Example 1. Initial use. Entry at $0100. Shows initial entries and stops with an exclamation point (!).*

```
$J 00AD
FUNCTION (N,B,Z,L,F,E,A,S,D) ? N

DAWSON; HAROLD *        453 287 5496  310427
4730 FULTON RD.          JACKSON; MI        76321


JACKSON; JOHN           301 251 6486  290403
243 AMOS ST.             NEWBURG; MO        33557


YARREL; PHYLLIS         474 976 3287  340823
455 53RD ST.             SAN FRANCISCO; CA 34567
```

*Example 2. Normal operation. Entry at $00AD. Name sort.*

- N – Sort by name
- Z – Sort by zip code
- F – List "flagged" entries
- A – Add an entry
- D – Delete an entry by sub-command as under S
- B – Sort by birth year
- L – List as currently sorted
- E – Exit to Monitor (or File Manager)
- S – Select an entry by sub-command:
  - # – name (#name#)
  - B – birth month (B04B)
  - Z – zip code (Z20783Z)

*Table 1.*

*C.H. Looney*
*3406 Notre Dame Street*
*Hyattsville, MD 20783*

This machine-language program is for inserting, retrieving, sorting and deleting name, address and birth date information.

The program takes about 1200 bytes of storage. At least 80 entries can be stored in an 8K machine.

Sorting is done by the primitive "bubble" method. It takes about one second to sort 25 entries and about four seconds to sort 50 entries. Sorting can be done by name, birth year or zip code.

Entries can be selected for printing or viewing on a CRT terminal by name, birth month, zip code or "flag." The "flagged" entries are marked by an asterisk after the name. Entries can be deleted after selection by name, birth month or zip code.

Function selection is by single letter input, except for those operations involving selection of a sort field (name, birth month or zip code). The sort field can be as short or as long as desired.

The functions are listed in Table 1.

The program (Listing 1) is particularly valuable to 6800 users without disk capability.

The first section of the program, from 00A3 to 00FF, is an executive to accept commands and branch to the proper areas of the program to accomplish the desired results. Start the program at $0100 the first time it is used to set up memory space and get first entries. Succeeding uses should enter at $00AD (EXEC) to add entries, sort, print, delete entries and so on.

The listing was prepared with a disassembler based on one described by Bob Lentz in the May 1979 *Byte*. His disassembler shows jump and branch destinations by address; I have included the capability of adding more conventional name destinations and labels along with comments. You may find the address destinations more helpful in understanding the operation of the program.

My interest in a name/address data-base program was whetted by an article in the October 1978 *Kilobaud* by Wantz and Bateman ("A Useful Address List Program," p. 102). They wrote their program in BASIC, however, and thus took a large amount of memory, leaving only room for about 15 entries in my 8K memory. This program provides space for more than 80 entries in that 8K of memory and sorts the entries in an acceptable time. Ambitious enthusiasts may wish to put in a more advanced sorting program, for both exercise and improvement.

The initialization section is the second

```
FUNCTION (N,B,Z,L,F,E,A,S,D) ? Z

JACKSON; JOHN          301 251 6486  290403
243 AMOS ST.             NEWBURG; MO        33557

YARREL; PHYLLIS        474 976 3287  340823
455 53RD ST.             SAN FRANCISCO; CA 34567

DAWSON; HAROLD *       453 287 5496  310427
4730 FULTON RD.          JACKSON; MI        76321
```

*Example 3. Birth year sort.*

```
FUNCTION (N,B,Z,L,F,E,A,S,D) ? B

JACKSON; JOHN          301 251 6486  290403
243 AMOS ST.             NEWBURG; MO        33557

DAWSON; HAROLD *       453 287 5496  310427
4730 FULTON RD.          JACKSON; MI        76321

YARREL; PHYLLIS        474 976 3287  340823
455 53RD ST.             SAN FRANCISCO; CA 34567
```

*Example 4. Zip code sort.*

```
FUNCTION (N,B,Z,L,F,E,A,S,D) ? F

DAWSON; HAROLD *       453 287 5496  310427
4730 FULTON RD.          JACKSON; MI        76321
```

*Example 5. Flag sort.*

```
FUNCTION (N,B,Z,L,F,E,A,S,D) ? S #YAR#

YARREL; PHYLLIS        474 976 3287  340823
455 53RD ST.             SAN FRANCISCO; CA 34567
```

*Example 6. Selection of name by first three letters.*

```
FUNCTION (N,B,Z,L,F,E,A,S,D) ? S B04B

JACKSON; JOHN          301 251 6486  290403
243 AMOS ST.             NEWBURG; MO        33557

DAWSON; HAROLD *       453 287 5496  310427
4730 FULTON RD.          JACKSON; MI        76321
```

*Example 7. Selection by birth month (April—04)*

```
FUNCTION (N,B,Z,L,F,E,A,S,D) ? D Z76321Z

DELETE THIS ENTRY (Y/N)?

DAWSON; HAROLD *       453 287 5496  310427
4730 FULTON RD.          JACKSON; MI        76321

Y
FUNCTION (N,B,Z,L,F,E,A,S,D) ? N

JACKSON; JOHN          301 251 6486  290403
243 AMOS ST.             NEWBURG; MO        33557

YARREL; PHYLLIS        474 976 3287  340823
455 53RD ST.             SAN FRANCISCO; CA 34567

FUNCTION (N,B,Z,L,F,E,A,S,D) ? E
CFM/3 VER 2.7
```

*Example 8. Deletion by zip code, name sort, return to Manager.*

part of the program. It sets up the area for storage of data from $0500 to $1310, allowing for 45 entries. Additional space can be allocated at $50 (80 decimal) spaces per entry.

The third section, the interrupt routine starting at $016A, permits stopping the printer on control-S and restarting on control-Q. This is particularly useful on a CRT screen; information can be jotted down as the entire data base is scanned. The interrupt routine also permits returning to the executive by typing in control-C either during printing or during a pause.

The listing section, $019F through $01DD, sets up field lengths and branches to the interrupt routine to print the entries. LIST is a subroutine called by the executive and other program sections to print entries one at a time. SWTBUG routines are used throughout this program for input (INCH-E1AC), output (OUTCH-E1D1), printing a space (OUTS-E0CC) and printing a string ending with $04 (PDATA1-E07E).

The sorting routines starting at $01DE are based on ideas in *"6800" Software Gourmet Guide & Cook Book* by Scelbi Computer Consulting, Inc. This book is ideal for the beginning 6800 user and is also useful for the expert to use as a handbook and reference. The sorting routines use the primitive bubble sort, but with the relatively small number of entries, sorting times are not long. The first part of this section simply

*Listing 1.*

```
*****  EXECutive routine - jumps or branches to routines on
                            basis of input commands

00A3 81 44    CONTEX  CMP A #44    #'D         is it a D?
00A5 26 06            BNE 00AD     EXEC        if not, go to EXEC
00A7 7E 03B1          JMP 03B1     DELETE      else, go DELETE
00AA 7E 1780  JCFM    JMP 1780     CFM/3       go to File Manager
00AD BD 0300  EXEC    JSR 0300     PCRLF       print CR & LF
00B0 CE 04BD          LDX #04BD    #'FUNCTION... load menu index
00B3 BD E07E          JSR E07E     PDATA1      and print menu
00B6 BD E1AC          JSR E1AC     INCH        get a character
00B9 81 4E            CMP A #4E    #'N         is it an N?
00BB 27 1F            BEQ 00DC     JSORTN      if so, go sort by name
00BD 81 42            CMP A #42    #'B         is it a B?
00BF 27 20            BEQ 00E1     JSORTB      if so, go sort by birth year
00C1 81 5A            CMP A #5A    #'Z         is it a Z?
00C3 27 21            BEQ 00E6     JSORTZ      if so, go sort by zip codes
00C5 81 4C            CMP A #4C    #'L         is it an L?
00C7 27 20            BEQ 00E9     JLIST       if so, list entries
00C9 81 46            CMP A #46    #'F         is it an F?
00CB 27 2B            BEQ 00F8     JFLAG       if so, list flagged entries
00CD 81 45            CMP A #45    #'E         is it an E?
00CF 27 D9            BEQ 00AA     JCFM        if so, go to Manager
00D1 81 41            CMP A #41    #'A         is it an A?
00D3 27 36            BEQ 010B     RESTRT      if so, go add an entry
00D5 81 53            CMP A #53    #'S         is it an S?
00D7 26 CA            BNE 00A3     CONTEX      if not, go to EXEC
00D9 7E 038C          JMP 038C     SELECT      else, select an entry
00DC BD 01DE  JSORTN  JSR 01DE     SORTN       go sort by names
00DF 20 08            BRA 00E9     JLIST       and list entries
00E1 BD 020C  JSORTB  JSR 020C     SORTB       go sort by birth year
00E4 20 03            BRA 00E9     JLIST       and list entries
00E6 BD 01F5  JSORTZ  JSR 01F5     SORTZ       go sort by zip codes
00E9 BD 0300  JLIST   JSR 0300     PCRLF       print CR & LF
00EC CE 0500          LDX #0500    #DATA       load data index
00EF BD 019F          JSR 019F     LIST        and list the entries
00F2 6D 01            TST X 01                 through listing?
00F4 26 F9            BNE 00EF     JLIST+6     if not, continue
00F6 20 B5    JEXEC   BRA 00AD     EXEC        else, go to EXEC
00F8 BD 0300  JFLAG   JSR 0300     PCRLF       print CR & LF
00FB BD 0276          JSR 0276     FLAG        and list flagged entries
00FE 20 F6            BRA 00F6     JEXEC       go to EXEC

*****  START routine - clears desired memory area and
                        adds entries to data base

0100 CE 0500  START   LDX #0500    #DATA       load data index
0103 6F 00            CLR X 00                 and clear memory location
0105 08               INX                      go to next location
0106 8C 1311          CPX #1311    #DATEND+1 end of data memory?
```

```
0109 26 F8            BNE 0103    START+3    if not, do it again
010B CE 0500  RESTRT  LDX #0500   #DATA      load data index
010E BD 02E9          JSR 02E9    FNDEND     move to end of entries
0111 BD 0300          JSR 0300    PCRLF      print CR & LF
0114 86 0D            LDA A #0D   #'CR       load a CR
0116 A7 00            STA A X 00             and store it
0118 08               INX                    go to next location
0119 C6 14            LDA B #14              name field length
011B 8D 16            BSR 0133    PROMPT     go get a name
011D C6 0A            LDA B #0A              telephone number field length
011F 8D 12            BSR 0133    PROMPT     go get the number
0121 C6 06            LDA B #06              birthdate field length
0123 8D 0E            BSR 0133    PROMPT     go get the date
0125 C6 15            LDA B #15              street field length
0127 8D 0A            BSR 0133    PROMPT     go get the address
0129 C6 11            LDA B #11              city/state field length
012B 8D 06            BSR 0133    PROMPT     go get the city/state
012D C6 05            LDA B #05              zip code field length
012F 8D 02            BSR 0133    PROMPT     go get the zip
0131 20 D8            BRA 010B    RESTRT     go back for another one
0133 DF 38    PROMPT  STX D 38    SAVEX1     save the index
0135 C1 14            CMP B #14              is it the name field?
0137 26 03            BNE 013C               if not, check next
0139 CE 0417          LDX #0417   #'NAME...  else load name index
013C C1 0A            CMP B #0A              is it the telephone field?
013E 26 03            BNE 0143               if not, check next
0140 CE 0431          LDX #0431   #'TEL...   else load telephone index
0143 C1 06            CMP B #06              is it the birthdate field?
0145 26 03            BNE 014A               if not, check next
0147 CE 044F          LDX #044F   #'BIRTH...else load birthdate index
014A C1 15            CMP B #15              is it the street field?
014C 26 03            BNE 0151               if not, check next
014E CE 0468          LDX #0468   #'STREET... else load street index
0151 C1 11            CMP B #11              is it the city/state field?
0153 26 03            BNE 0158               if not, check next
0155 CE 0484          LDX #0484   #'CITY ... else load city/state index
0158 C1 05            CMP B #05              is it the zip code field?
015A 26 03            BNE 015F               if not, go print
015C CE 04A4          LDX #04A4   #'ZIP...   else load zip index
015F BD E07E          JSR E07E    PDATA1     print message
0162 DE 38            LDX D 38    SAVEX1     reload index
0164 BD 0300          JSR 0300    PCRLF      print CR & LF
0167 7E 029B          JMP 029B    INPUT      now go get the keyboard input

***** Interrupt routine - permits stopping and restarting the
                          print routine or returning to EXEC

016A B6 8004  INTRUP  LDA A 8004             look at key board address
016D 47               ASR A                  any action?
016E 24 1F            BCC 018F    CONTP      if not, continue printing
0170 B6 8005          LDA A 8005             else load keyboard entry
0173 81 03            CMP A #03   #'↑C       is it a Control-C?
0175 26 03            BNE 017A               if not, check for another
0177 7E 00AD  JEXEC1  JMP 00AD    EXEC       else return to EXEC
017A 81 13            CMP A #13   #'↑S       is it a Control-S?
017C 26 11            BNE 018F    CONTP      if not, continue printing
017E B6 8004  LOOP    LDA A 8004             look at keyboard address
0181 47               ASR A                  any action?
0182 24 FA            BCC 017E    LOOP       if not, continue looping
0184 B6 8005          LDA A 8005             else load keyboard entry
0187 81 03            CMP A #03   #'↑C       is it a Control-C?
0189 27 EC            BEQ 0177    JEXEC      if so, return to EXEC
018B 81 11            CMP A #11   #'↑Q       is it a Control-Q?
018D 26 EF            BNE 017E    LOOP       if not, continue looping
018F A6 00    CONTP   LDA A X 00             get character from memory
0191 81 00            CMP A #00              is it 00?
0193 26 02            BNE 0197    OUTC       if not, go print it
0195 86 20            LDA A #20   #'SP       else load a space
0197 BD E1D1  OUTC    JSR E1D1    OUTCH      and print it
019A 08               INX                    go to next memory location
019B 5A               DEC B                  decrement the counter
019C 26 CC            BNE 016A    INTRUP     if not through, get another
019E 39               RTS                    else return
***** LIST routine - sets up field counters and branches to
                     INTRUP to list name, birthdate, telhone
                     number and address information

019F C6 15    LIST    LDA B #15              CR + name field length
01A1 8D C7            BSR 016A    INTRUP     go print name
01A3 BD E0CC          JSR E0CC    OUTS       print a space
01A6 C6 03            LDA B #03              tel. area code field length
01A8 8D C0            BSR 016A    INTRUP     go print area code
01AA BD E0CC          JSR E0CC    OUTS       print a space
01AD C6 03            LDA B #03              tel. exchange field length
01AF 8D B9            BSR 016A    INTRUP     go print exchange
01B1 BD E0CC          JSR E0CC    OUTS       print a space
01B4 C6 04            LDA B #04              tel. number field length
01B6 8D B2            BSR 016A    INTRUP     go print number
01B8 BD E0CC          JSR E0CC    OUTS       print a space
01BB BD E0CC          JSR E0CC    OUTS       print another space
01BE C6 06            LDA B #06              birthdate field length
01C0 8D A8            BSR 016A    INTRUP     go print birthdate
01C2 BD 0300          JSR 0300    PCRLF      print CR & LF
01C5 C6 15            LDA B #15              street field length
01C7 8D A1            BSR 016A    INTRUP     go print street address
01C9 BD E0CC          JSR E0CC    OUTS       print a space
01CC C6 11            LDA B #11              city/state field length
01CE 8D 9A            BSR 016A    INTRUP     go print city/state
01D0 BD E0CC          JSR E0CC    OUTS       print a space
01D3 C6 05            LDA B #05              zip code field length
01D5 8D 93            BSR 016A    INTRUP     go print zip
01D7 BD 0300          JSR 0300    PCRLF      print CR & LF
01DA BD 0300          JSR 0300    PCRLF      print CR & LF
01DD 39               RTS                    and return
```

sets up field lengths and starting locations in order to ease the user's problems in directing sorting operations.

The flag routine ($0276 through $0299) searches for the flag (an asterisk—*) and uses LIST to print selected entries. The flag need not be at any particular location as long as it is in the name field. It could precede names for selection, be buried in the name or follow the names to be selected.

The input routine starting at $029B is used to format the input data, to recognize when input is completed and to permit back-space correction or entry canceling. The exclamation point (!) is used to stop entry and return to the executive. It should be used as the first character in an entry field; otherwise, a partial entry will be made.

Telephone numbers, birth dates and zip codes are expected to fill designated fields, while names and addresses may be truncated by using a comma as the final character. The comma can also be used to blank a field when the data is unknown by typing the comma as the first character in that field. If a field is overrun, the overflow characters will fall into the next field; therefore, you must be careful when filling entries.

Since a comma is used to truncate names and addresses, a semicolon should be used to separate city from state and last name from initials. You can use a return (CR) to terminate an entry and cause the program to consider that entry complete and prepare for the next entry.

The two routines next in line simply print a CR and LF and advise you when the last entry space has been filled.

The rest of the program is based on the excellent ideas put forth by Peter Stark's article on a BASIC editor (see "An Editor for 6800 BASIC Programs," January 1979 *Kilobaud Microcomputing*, p. 22). In particular, his search and replace routines have been adapted to the needs of this data-base program.

SELECT and DELETE are the two primary routines. Each uses DELIM to set up a delimiter character and READST to set up a string for matching to names, birth months or zip codes in the stored data. The delimiter character must be a number sign (#) for use in selecting names, B for choosing birth months and Z for selecting zip codes.

SELECT will print out as many entries as have matching characteristics with the string. DELETE will ask you whether you wish to delete an entry chosen to match the string. If you do not accept the offered entry, DELETE will go back to get another until all matching entries have been offered.

Listing 2 is a complete machine-code listing of the program followed by a decoded printout of the character strings used as messages. Examples 1–8 show the use of the data-base program. ■

```
***** SORT routines - name, birth year and zip code sort routines

01DE CE 022A  SORTN   LDX #022A              index for sort field length
01E1 86 14            LDA A #14              name sort field length
01E3 A7 00            STA A X 00
01E5 86 01            LDA A #01              name field start location
01E7 A7 09            STA A X 09
01E9 86 51            LDA A #51              next name field start
01EB A7 0B            STA A X 0B
01ED 86 3C            LDA A #3C              distance to next name field
01EF A7 13            STA A X 13
01F1 A7 42            STA A X 42
01F3 20 2C            BRA 0221     SORT      go do the sorting
01F5 CE 022A  SORTZ   LDX #022A              index for sort field length
01F8 86 05            LDA A #05              zip code field length
01FA A7 00            STA A X 00
01FC 86 4B            LDA A #4B              zip code field start location
01FE A7 09            STA A X 09
0200 86 9B            LDA A #9B              location of next zip
0202 A7 0B            STA A X 0B
0204 86 4B            LDA A #4B              distance to next zip
0206 A7 13            STA A X 13
0208 A7 42            STA A X 42
020A 20 15            BRA 0221     SORT      go do the sorting
020C CE 022A  SORTB   LDX #022A              index for sort field length
020F 86 06            LDA A #06              birthdate field length
0211 A7 00            STA A X 00
0213 86 1F            LDA A #1F              birthdate field start location
0215 A7 09            STA A X 09
0217 86 6F            LDA A #6F              location of next birthdate
0219 A7 0B            STA A X 0B
021B 86 4A            LDA A #4A              distance to next birthdate
021D A7 13            STA A X 13
021F A7 42            STA A X 42
0221 CE 0500  SORT    LDX #0500    #DATA     load data index
0224 8C 12C0  NEREND  CPX #12C0    #DATEND-$50 index of last possible entry
0227 27 06            BEQ 022F     SRTRET    return if at last entry
0229 C6 14            LDA B #14              sort field length
022B 6D 51            TST X 51               check to see if at data end
022D 26 01            BNE 0230     CKNEXT    if not, continue
022F 39       SRTRET  RTS                    else return
0230 DF 38    CKNEXT  STX D 38     SAVEX1    save index
0232 A6 01            LDA A X 01             load character
0234 A1 51            CMP A X 51             compare with next field
0236 26 0C            BNE 0244     CKGTLT    if not same, see which larger
0238 08               INX                    else move on
0239 5A               DEC B                  and decrement counter
023A 26 F6            BNE 0232     CKNEXT+2  continue if not at field end
023C C6 3C            LDA B #3C              else go to next field
023E 08               INX                    by advancing step
023F 5A               DEC B                  by step
0240 26 FC            BNE 023E               keep at it till at next field
0242 20 E0            BRA 0224     NEREND    and sort next pair of fields
0244 23 21    CKGTLT  BLS 0267     FINEND    if less go to field end
0246 C6 50            LDA B #50              else bubble sort them
0248 DE 38            LDX D 38     SAVEX1    by saving the index
024A A6 00    NOTYET  LDA A X 00             and interchanging characters
024C 97 35            STA A D 35   TEMP      between DATA and TEMP locations
024E A6 50            LDA A X 50             until they're all
0250 A7 00            STA A X 00             interchanged
0252 96 35            LDA A D 35   TEMP
0254 A7 50            STA A X 50
0256 08               INX
0257 5A               DEC B
0258 26 F0            BNE 024A     NOTYET    keep at it till done
025A C6 A0            LDA B #A0              and then load the counter
025C 09       NMINUS  DEX                    to backup two entries
025D 5A               DEC B                  now go back to last entry
025E 26 FC            BNE 025C     NMINUS    to see if it needs to move
0260 8C 0500          CPX #0500    #DATA     but don't go too far
0263 2D BC            BLT 0221     SORT      if so, go back and sort again
0265 20 BD            BRA 0224     NEREND    else continue from here
0267 08       FINEND  INX                    now go to field end
0268 5A               DEC B                  step by step
0269 26 FC            BNE 0267     FINEND    there yet?
026B C6 3C            LDA B #3C              ok, now go to next sort field
026D 08               INX                    step by step
026E 5A               DEC B
026F 26 FC            BNE 026D               once you're there
0271 20 B1            BRA 0224     NEREND    sort the next pair of fields
0273 7E 010B  JRES    JMP 010B     RESTRT    used as a jump island

***** FLAG routine - to select and print flagged entries

0276 CE 0500  FLAG    LDX #0500    #DATA     data start index
0279 C6 15            LDA B #15              CR + name field length
027B A6 00            LDA A X 00             get a character from memory
027D 81 2A            CMP A #2A    #'*       is it the flag (*)?
027F 27 0F            BEQ 0290     BACK      if so, go back up
0281 08               INX                    if not, go to next character
0282 5A               DEC B                  decrement the counter
0283 26 F6            BNE 027B               if not at field end, do again
0285 C6 3B            LDA B #3B              else go to next field
0287 08               INX                    step by step
0288 5A               DEC B                  at end yet?
0289 26 FC            BNE 0287               if not, do it again
028B A6 00            LDA A X 00             then get a character
028D 26 EA            BNE 0279     FLAG+3    if not 00, check this field
028F 39               RTS                    else return
0290 09       BACK    DEX                    back up to start of
0291 5C               INC B                  name field
0292 C1 15            CMP B #15              through yet?
0294 26 FA            BNE 0290     BACK      if not, keep backing
0296 BD 019F          JSR 019F     LIST      else go list entry
0299 20 DE            BRA 0279     FLAG+3    and look for next flag
```

```
***** INPUT routine - accepts input data and puts in correct place

029B BD E1AC  INPUT   JSR E1AC    INCH        get character from keyboard
029E 81 21            CMP A #21   #'!         is it an !?
02A0 26 03            BNE 02A5                if not, continue
02A2 7E 00AD          JMP 00AD    EXEC        else go to EXEC
02A5 81 0F            CMP A #0F   #'↑O        is it Control-O?
02A7 26 10            BNE 02B9    CONT        if not, continue
02A9 CE 0500          LDX #0500   #DATA       else load DATA index
02AC 8D 3B            BSR 02E9    FNDEND      find end of entries
02AE C6 50            LDA B #50               set counter for entry space
02B0 09       BACKUP  DEX                     and backup to start
02B1 5A               DEC B                   of current entry
02B2 26 FC            BNE 02B0    BACKUP      if not done, do again
02B4 6F 00            CLR X 00                clear first location
02B6 7E 0111          JMP 0111    RESTRT+6    and start entry again
02B9 81 0D    CONT    CMP A #0D   #'CR        is it a CR?
02BB 27 B6            BEQ 0273    JRES        if so, go back to RESTRT
02BD 81 08            CMP A #08   #'BS        is it a BS?
02BF 26 0E            BNE 02CF    CONT1       if not, continue
02C1 09               DEX                     else backup one
02C2 5C               INC B                   space, and
02C3 36               PSH A                   save ACC A
02C4 A6 00            LDA A X 00              get a character
02C6 81 0D            CMP A #0D   #'CR        is it a CR?
02C8 32               PUL A                   restore ACC A
02C9 26 D0            BNE 029B    INPUT       if not CR, get keyboard entry
02CB 08               INX                     else go forward
02CC 5A               DEC B                    a space
02CD 20 CC            BRA 029B    INPUT       and get the entry
02CF 81 2C    CONT1   CMP A #2C   #',         is it a ,?
02D1 26 0A            BNE 02DD    CONT2       if not, continue
02D3 86 00    HAVECM  LDA A #00               else load 00
02D5 A7 00            STA A X 00              and store to fill
02D7 08               INX                     until field is full
02D8 5A               DEC B                   full field?
02D9 26 F8            BNE 02D3    HAVECM      if not, repeat
02DB 20 06            BRA 02E3    RETURN      else start back for another one
02DD A7 00    CONT2   STA A X 00              store the character
02DF 08               INX                     and bump the index counter
02E0 5A               DEC B                   decrement counter; field full?
02E1 26 04            BNE 02E7    CONT3       if not, go CONT3
02E3 BD E0CC  RETURN  JSR E0CC    OUTS        else print a space
02E6 39               RTS                     and return
02E7 20 B2    CONT3   BRA 029B    INPUT       go for more input
02E9 6D 01    FNDEND  TST X 01                look at next character
02EB 27 0A            BEQ 02F7    ENDRET      if 00, return
02ED C6 50            LDA B #50               else load counter
02EF 08       ADVNCE  INX                     and move ahead
02F0 5A               DEC B                   step by step
02F1 26 FC            BNE 02EF    ADVNCE      if not through, keep advancing
02F3 8D 05            BSR 02FA    CKSPAC      go check if at end
02F5 20 F2            BRA 02E9    FNDEND      and see if at next empty spot
02F7 01       ENDRET  NOP
02F8 01               NOP
02F9 39               RTS                     go back where you came from
02FA 8C 1310  CKSPAC  CPX #1310   #DATEND     end of space (45 entries)
02FD 27 0C            BEQ 030B    MEMFUL      if at end, tell user
02FF 39               RTS                     else return

***** CR LF routine - prints a CR & LF

0300 DF 38    PCRLF   STX D 38    SAVEX1      store the index
0302 CE 0400          LDX #0400   #'CR LF...load message index
0305 BD E07E          JSR E07E    PDATA1      print the CR & LF
0308 DE 38            LDX D 38    SAVEX1      restore index
030A 39               RTS                     and return

***** MEMory FULl Routine - prints MEMORY FULL when allocated
                            memory space is filled

030B CE 0406  MEMFUL  LDX #0406   #'MEM...    load message index
030E BD E07E          JSR E07E    PDATA1      print the message
0311 7E 1780          JMP 1780    CFM/3       and return to Manager

***** READ STring routine - accepts keyboard input of string to be
                            found in DATA

0314 BD E1AC  READST  JSR E1AC    INCH        get keyboard input
0317 11               CBA                     compare with ACC B
0318 27 09            BEQ 0323    EXITRS      if equal, return
031A 81 0D            CMP A #0D   #'CR        is it a CR?
031C 27 05            BEQ 0323    EXITRS      if so, return
031E A7 00            STA A X 00              else store the character
0320 08               INX                     bump the index and
0321 20 F1            BRA 0314    READST      go back for another input
0323 39       EXITRS  RTS                     return

***** FIND STring routine - searches for STRING; if found returns
                            with ACC B = 1

0324 96 40    FINDST  LDA A D 40  STRING-1    get the delimiter
0326 81 23            CMP A #23   #'#         is it a #?
0328 27 16            BEQ 0340    NFIND       if so, go looking for a name
032A 81 5A            CMP A #5A   #'Z         is it a Z?
032C 27 0C            BEQ 033A    ZFIND-2     if so, go looking for a zip
032E 81 42            CMP A #42   #'B         is it a B?
0330 26 67            BNE 0399    JEXEC2      if not, go to EXEC
***** SELECT routine - prints entries based on user selection
                       of name, zip code or birthdate

038C 8D EC    SELECT  BSR 037A    DELIM       get delimiter and STRING
038E CE 0500          LDX #0500   #DATA       load DATA index
0391 BD 0300          JSR 0300    PCRLF       print CR & LF
```

# THE NATIONAL COMPUTER SHOWS

# HAVE WE GOT A PROGRAM FOR YOU!

The new computers are showing off.

Over $50 million worth of equipment in over 100,000 square feet of space, including the latest software and hardware for business, government, home and personal use. Everything the NCC show has and more will be on display, and you can buy it all right on the spot.

Computers costing $150 to $250,000, mini and micro computers, data- and word-processing equipment, telecommunications, office machines, peripheral equipment and services from leading names in the industry like IBM, Xerox, Radio Shack and Apple will all be there.

There'll be conferences on business uses of small to medium sized computers, and how to make purchasing evaluations.

There'll be robots, computerized video games, computer art and computer music.

Everyone from kids to people who earn their living with computers will have a great time at the largest computer show ever organized in each region.

Admission for adults is $5. The public is invited, and no pre-registration is necessary.

Don't miss the computer show that mixes business with pleasure. Show up for the show.

## THE MID-ATLANTIC COMPUTER SHOW

**WASHINGTON, D.C.**
D.C. ARMORY/STARPLEX
THURSDAY-SUNDAY
SEPTEMBER 18-21
11 A.M. TO 9 P.M. THURS.-SAT.
11 A.M. TO 5 P.M. SUN.

## THE MID-WEST COMPUTER SHOW

**CHICAGO**
McCORMICK PLACE
THURSDAY-SUNDAY
OCTOBER 16-19
11 A.M. TO 9 P.M. THURS.-SAT.
11 A.M. TO 5 P.M. SUN.

## THE NORTHEAST COMPUTER SHOW

**BOSTON**
HYNES AUDITORIUM
PRUDENTIAL CENTER
THURSDAY-SUNDAY
NOVEMBER 20-23
11 A.M. TO 9 P.M. THURS.-SAT.
11 A.M. TO 5 P.M. SUN.

Produced by National Computer Shows,
824 Boylston Street, Chestnut Hill, MA 02167.
Telephone (617) 739-2000.

Please send me:

☐ ____ adult tickets at $5 each. I have enclosed the proper amount of $ ____

☐ Information on the show's conference program.

☐ Hotel registration information ☐ Exhibitor rental information

Please print: Name ____________

Address ____________

City ____________ State ____________ Zip ____________

```
0394 8D 8E     FLOOP    BSR 0324    FINDST     go find the STRING
0396 5D                 TST B                  find one?
0397 26 03              BNE 039C    LOOKCR     if so, go back to CR
0399 7E 00AD   JEXEC2   JMP 00AD    EXEC       else go EXEC
039C 09        LOOKCR   DEX                    start backing up
039D A6 00              LDA A X 00             get a character
039F 81 0D              CMP A #0D   #'CR       is it a CR?
03A1 26 F9              BNE 039C    LOOKCR     if not, keep backing
03A3 BD 019F            JSR 019F    LIST       else print entry
03A6 6D 01              TST X 01               is it the last one?
03A8 27 EF              BEQ 0399    JEXEC2     if so, go EXEC
03AA 8C 1310            CPX #1310   #DATEND    at end of memory?
03AD 26 E5              BNE 0394    FLOOP      if not, look for another
03AF 20 E8              BRA 0399    JEXEC2     else go EXEC

***** DELETE routine - deletes one entry based upon user's choice
                       of name, zip or birthdate; gets user
                       confirmation before deleting entry

03B1 BD 037A   DELETE   JSR 037A    DELIM      get delimiter and STRING
03B4 CE 0500            LDX #0500   #DATA      load DATA index
03B7 BD 0324   RPTDEL   JSR 0324    FINDST     find the STRING
03BA 5D                 TST B                  find one?
03BB 26 02              BNE 03BF    LOOKC      if so, back up to a CR
03BD 20 DA              BRA 0399    JEXEC2     if not, go EXEC
03BF 09        LOOKC    DEX                    start backing up
03C0 A6 00              LDA A X 00             get a character
03C2 81 0D              CMP A #0D   #'CR       is it a CR?
03C4 26 F9              BNE 03BF    LOOKC      if not, keep backing
03C6 BD 0300            JSR 0300    PCRLF      print CR & LF
03C9 CE 04DD            LDX #04DD   #'DEL...   load message index
03CC BD E07E            JSR E07E    PDATA1     and print message
03CF DE 38              LDX D 38    SAVEX1     save the index
03D1 BD 019F            JSR 019F    LIST       print the entry
03D4 BD E1AC            JSR E1AC    INCH       get a character from keyboard
03D7 81 59              CMP A #59   #'Y        is it a Y?
03D9 27 0B              BEQ 03E6    DELET1     if so, go delete the entry
03DB 6D 01              TST X 01               last entry?
03DD 27 BA              BEQ 0399    JEXEC2     if so, go EXEC
03DF 8C 1310            CPX #1310   #DATEND    end of memory?
03E2 26 D3              BNE 03B7    RPTDEL     if not, check another entry
03E4 20 B3              BRA 0399    JEXEC2     else go EXEC
03E6 09        DELET1   DEX                    start backing up
03E7 A6 00              LDA A X 00             load a character
03E9 81 0D              CMP A #0D   #'CR       is it a CR?
03EB 26 F9              BNE 03E6    DELET1     if not, keep backing
03ED 86 5B              LDA A #5B   #'[        load a [
03EF A7 01              STA A X 01             and store as 1st entry char.
03F1 BD 01DE            JSR 01DE    SORTN      go SORTN to put this entry last
03F4 C6 50              LDA B #50              load counter with entry size
03F6 08        CLEAR    INX                    bump the index
03F7 6F 00              CLR X 00               clear a character
03F9 5A                 DEC B                  decrement the counter; at end?
03FA 26 FA              BNE 03F6    CLEAR      if not, do again
03FC 20 9B              BRA 0399    JEXEC2     when done, go EXEC
0332 C6 20              LDA B #20              load the counter to get to
0334 08        BFIND    INX                    birthdate location and move it
0335 5A                 DEC B                  there - through?
0336 26 FC              BNE 0334    BFIND      if not, keep moving
0338 20 06              BRA 0340    NFIND      else go looking
033A C6 4A              LDA B #4A              load the counter to get to
033C 08        ZFIND    INX                    zip code location and move it
033D 5A                 DEC B                  there - through?
033E 26 FC              BNE 033C    ZFIND      if not, keep moving
0340 08        NFIND    INX                    now move one more space
0341 A6 00              LDA A X 00             load a character
0343 91 41              CMP A D 41  STRING     match 1st STRING character?
0345 26 25              BNE 036C    NOTF       if not, go housekeep and return
0347 DF 3A     FOUND1   STX D 3A    SAVEX      else save index here
0349 DF 3C              STX D 3C    TEMPSO     and here
034B CE 0041            LDX #0041   #STRING    then load STRING index
034E DF 3E              STX D 3E    TEMPST     and store it here
0350 DE 3C     FLOOP1   LDX D 3C    TEMPSO     now load "data" index
0352 A6 00              LDA A X 00             and put the 1st char. in ACC A
0354 08                 INX                    bump the index
0355 DF 3C              STX D 3C    TEMPSO     and store new index
0357 DE 3E              LDX D 3E    TEMPST     load STRING index
0359 E6 00              LDA B X 00             and put its 1st char. in ACC B
035B 08                 INX                    bump its index
035C DF 3E              STX D 3E    TEMPST     and store it
035E 11                 CBA                    do characters match?
035F 26 09              BNE 036A    NOTF-2     if not, go housekeep and return
0361 9C 36              CPX A D 36  SPOIN      else see if at end of STRING
0363 26 EB              BNE 0350    FLOOP1     if not compare next characters
0365 DE 3A              LDX D 3A    SAVEX      else load original index
0367 C6 01              LDA B #01              set ACC B to 1 and
0369 39                 RTS                    return

***** NOT Found routine - resets index and sets ACC B to 0
                          if STRING not found

036A DE 3A              LDX D 3A    SAVEX      load index
036C 08        NOTF     INX                    and start looking for a CR
036D A6 00              LDA A X 00             get a character
036F 81 0D              CMP A #0D   #'CR       is it a CR?
0371 26 F9              BNE 036C    NOTF       if not, look again
0373 6D 01              TST X 01               is next character a 00?
0375 26 AD              BNE 0324    FINDST     if not, look for STRING again
0377 C6 00              LDA B #00              else set ACC B to 00
0379 39                 RTS                    and return

***** DELIMiter routine - gets and stores the delimiter character
                          used to select name, zip, or birthdate
                          information
```

```
037A CE 0040  DELIM   LDX #0040   #STRING-1 set index for delimiter
037D BD E0CC          JSR E0CC    OUTS      print a space
0380 BD E1AC          JSR E1AC    INCH      get the delimiter
0383 16               TAB                   put it in ACC B
0384 A7 00            STA A X 00            and in indexed location
0386 08               INX                   bump the index and
0387 8D 8B            BSR 0314    READST    go get STRING
0389 DF 36            STX D 36    SPOIN     store index for end of STRING
038B 39               RTS                   and return
```

*Listing 2.*

```
00A0 02 21 00 81 44 26 06 7E 03 B1 7E 17 80 BD 03 00
00B0 CE 04 BD BD E0 7E BD E1 AC 81 4E 27 1F 81 42 27
00C0 20 81 5A 27 21 81 4C 27 20 81 46 27 2B 81 45 27
00D0 D9 81 41 27 36 81 53 26 CA 7E 03 8C BD 01 DE 20
00E0 08 BD 02 0C 20 03 BD 01 F5 BD 03 00 CE 05 00 BD
00F0 01 9F 6D 01 26 F9 20 B5 BD 03 00 BD 02 76 20 F6
0100 CE 05 00 6F 00 08 8C 13 11 26 F8 CE 05 00 BD 02
0110 E9 BD 03 00 86 0D A7 00 08 C6 14 8D 16 C6 0A 8D
0120 12 C6 06 8D 0E C6 15 8D 0A C6 11 8D 06 C6 05 8D
0130 02 20 D8 DF 38 C1 14 26 03 CE 04 17 C1 0A 26 03
0140 CE 04 31 C1 06 26 03 CE 04 4F C1 15 26 03 CE 04
0150 68 C1 11 26 03 CE 04 84 C1 05 26 03 CE 04 A4 BD
0160 E0 7E DE 38 BD 03 00 7E 02 9B B6 80 04 47 24 1F
0170 B6 80 05 81 03 26 03 7E 00 AD 81 13 26 11 B6 80
0180 04 47 24 FA B6 80 05 81 03 27 EC 81 11 26 EF A6
0190 00 81 00 26 02 86 20 BD E1 D1 08 5A 26 CC 39 C6
01A0 15 8D C7 BD E0 CC C6 03 8D C0 BD E0 CC C6 03 8D
01B0 B9 BD E0 CC C6 04 8D B2 BD E0 CC BD E0 CC C6 06
01C0 8D A8 BD 03 00 C6 15 8D A1 BD E0 CC C6 11 8D 9A
01D0 BD E0 CC C6 05 8D 93 BD 03 00 BD 03 00 39 CE 02
01E0 2A 86 14 A7 00 86 01 A7 09 86 51 A7 0B 86 3C A7
01F0 13 A7 42 20 2C CE 02 2A 86 05 A7 00 86 4B A7 09
0200 86 9B A7 0B 86 4B A7 13 A7 42 20 15 CE 02 2A 86
0210 06 A7 00 86 1F A7 09 86 6F A7 0B 86 4A A7 13 A7
0220 42 CE 05 00 8C 12 C0 27 06 C6 14 6D 51 26 01 39
0230 DF 38 A6 01 A1 51 26 0C 08 5A 26 F6 C6 3C 08 5A
0240 26 FC 20 E0 23 21 C6 50 DE 38 A6 00 97 35 A6 50
0250 A7 00 96 35 A7 50 08 5A 26 F0 C6 A0 09 5A 26 FC
0260 8C 05 00 2D BC 20 BD 08 5A 26 FC C6 3C 08 5A 26
0270 FC 20 B1 7E 01 0B CE 05 00 C6 15 A6 00 81 2A 27
0280 0F 08 5A 26 F6 C6 3B 08 5A 26 FC A6 00 26 EA 39
0290 09 5C C1 15 26 FA BD 01 9F 20 DE BD E1 AC 81 21
02A0 26 03 7E 00 AD 81 0F 26 10 CE 05 00 8D 3B C6 50
02B0 09 5A 26 FC 6F 00 7E 01 11 81 0D 27 B6 81 08 26
02C0 0E 09 5C 36 A6 00 81 0D 32 26 D0 08 5A 20 CC 81
02D0 2C 26 0A 86 00 A7 00 08 5A 26 F8 20 06 A7 00 08
02E0 5A 26 04 BD E0 CC 39 20 B2 6D 01 27 0A C6 50 08
02F0 5A 26 FC 8D 05 20 F2 01 01 39 8C 13 10 27 0C 39
0300 DF 38 CE 04 00 BD E0 7E DE 38 39 CE 04 06 BD E0
0310 7E 7E 17 80 BD E1 AC 11 27 09 81 0D 27 05 A7 00
0320 08 20 F1 39 96 40 81 23 27 16 81 5A 27 0C 81 42
0330 26 67 C6 20 08 5A 26 FC 20 06 C6 4A 08 5A 26 FC
0340 08 A6 00 91 41 26 25 DF 3A DF 3C CE 00 41 DF 3E
0350 DE 3C A6 00 08 DF 3C DE 3E E6 00 08 DF 3E 11 26
0360 09 9C 36 26 EB DE 3A C6 01 39 DE 3A 08 A6 00 81
0370 0D 26 F9 6D 01 26 AD C6 00 39 CE 00 40 BD E0 CC
0380 BD E1 AC 16 A7 00 08 8D 8B DF 36 39 8D EC CE 05
0390 00 BD 03 00 8D 8E 5D 26 03 7E 00 AD 09 A6 00 81
03A0 0D 26 F9 BD 01 9F 6D 01 27 EF 8C 13 10 26 E5 20
03B0 E8 BD 03 7A CE 05 00 BD 03 24 5D 26 02 20 DA 09
03C0 A6 00 81 0D 26 F9 BD 03 00 CE 04 DD BD E0 7E DE
03D0 38 BD 01 9F BD E1 AC 81 59 27 0B 6D 01 27 BA 8C
03E0 13 10 26 D3 20 B3 09 A6 00 81 0D 26 F9 86 5B A7
03F0 01 BD.01 DE C6 50 08 6F 00 5A 26 FA 20 9B 01 01
0400 0D 0A 00 00 00 04 0D 0A 00 00 00 4D 45 4D 4F 52
0410 59 20 46 55 4C 4C 04 0D 0A 00 00 00 4E 41 4D 45
0420 20 28 32 30 20 43 48 41 52 41 43 54 45 52 53 29
0430 04 0D 0A 00 00 00 54 45 4C 45 50 48 4F 4E 45 20
0440 28 31 31 20 44 49 47 49 54 20 4D 41 58 29 04 0D
0450 0A 00 00 00 42 49 52 54 48 20 44 41 54 45 20 28
0460 59 59 4D 4D 44 44 29 04 0D 0A 00 00 00 53 54 52
0470 45 45 54 20 28 32 31 20 43 48 41 52 41 43 54 45
0480 52 53 29 04 0D 0A 00 00 00 43 49 54 59 3B 53 54
0490 41 54 45 20 28 31 37 20 43 48 41 52 41 43 54 45
04A0 52 53 29 04 0D 0A 00 00 00 5A 49 50 20 43 4F 44
04B0 45 20 28 35 20 44 49 47 49 54 53 29 04 46 55 4E
04C0 43 54 49 4F 4E 20 28 4E 2C 42 2C 5A 2C 4C 2C 46
04D0 2C 45 2C 41 2C 53 2C 44 29 20 3F 20 04 0D 0A 00
04E0 00 00 44 45 4C 45 54 45 20 54 48 49 53 20 45 4E
04F0 54 52 59 20 28 59 2F 4E 29 3F 0D 0A 00 00 00 04

0400                 MEMOR
0410 Y FULL           NAME
0420  (20 CHARACTERS)
0430          TELEPHONE
0440 (11 DIGIT MAX)
0450        BIRTH DATE (
0460 YYMMDD)          STR
0470 EET (21 CHARACTE
0480 RS)          CITY;ST
0490 ATE (17 CHARACTE
04A0 RS)          ZIP COD
04B0 E (5 DIGITS) FUN
04C0 CTION (N,B,Z,L,F
04D0 ,E,A,S,D) ?
04E0   DELETE THIS EN
04F0 TRY (Y/N)?
```

# Upgrading the Heath H8 With a Z-80 Microprocessor

*Now that Heath has boarded the Z-80 bandwagon, keep in step with the HZ8 adapter.*

*Patrick Swayne*
*290 Springdale*
*Sebastopol, CA 95472*

When the Heath Company decided to get into the hobby computer business, the 8080 was the most popular micro, and thus the most logical choice for their CPU. But the Z-80 has since eclipsed it in popularity, and now Heath has joined the parade by introducing their own Z-80-based machine, the H89. This computer uses the same disk and cassette operating systems as the H8. Currently, the only software available for it is 8080 software, but sooner or later software will appear that takes advantage of the Z-80's expanded instruction set. Over 15,000 H8 owners may be left behind as a result.

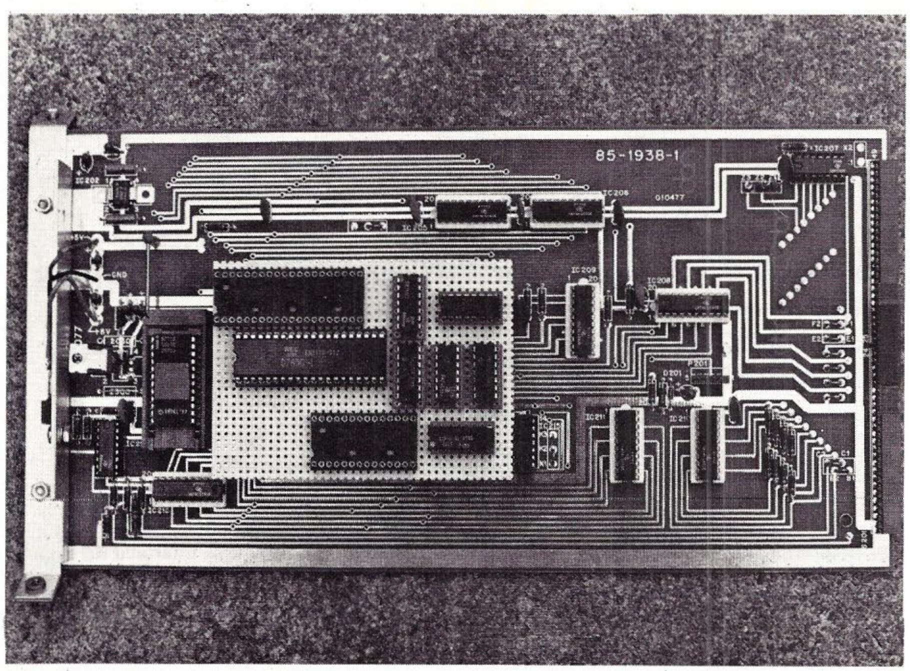



*Photo 1. H8 CPU board with the adapter plugged in. The Z-80 (D780C) and some TTL ICs replace the original 8080 and 8238. I placed two extra ICs for the optional section, which I designed after I took this photo, in the lower-left corner of the perfboard. The Textool socket to the left of the adapter is for testing homemade ROM monitors.*

Photo 1 shows my solution to this problem. It is a Z-80 adapter that mounts piggyback onto the H8 CPU board. It plugs into the sockets normally occupied by the 8080 CPU, the 8238 (or 74S438) system controller and the 8224 clock driver. No modifications are required on the CPU board itself, and at any time you can remove the adapter and replace the original ICs. It can be built for about $30 to $40, including the Z-80 (which has dropped to less than $15 at some mail-order houses).

**How the Adapter Works**

The schematic of the adapter is shown in Fig. 1. In this circuit, the original CPU and system controller are replaced by a Z-80, six commonly available TTL chips, a capacitor and some resistors. (Two of the TTLs and the capacitor may be eliminated if the optional section is not built.) The clock driver is also shown because some connections were made to it. (A socket for the clock driver was included in the actual assembly to make these connections possible, even though it is still electrically on the CPU board.)

This circuit effectively emulates all of the signals normally produced by the 8080 and 8238 that are used by the H8. This is easy to accomplish because Heath chose to use the fully decoded 8238 system controller signals, rather than the undecoded status or control signals (except M1) that are multiplexed onto the data bus of the 8080.

You can produce those signals by simply ANDing together the appropriate Z-80 outputs. For example, the $\overline{MEMR}$ (memory

read) signal is produced by ANDing the $\overline{MREQ}$ (memory request) and $\overline{RD}$ (read) outputs. The NOR gates on the schematic are used as negative AND gates in this application (except those used as inverters).

Noninverting OR gates (such as the 74LS32) could have been used, eliminating the need for inverters at the outputs of the NORs, but the chip count would not have been reduced, since the circuit requires at least five ORs and seven inverters. I chose to use 74LS02s because they are more readily available. Pull-up resistors are required at each of the four memory and I/O control outputs of the Z-80, because they are Tri-state and might, at some time, be undefined.

Deriving the 8238-type memory and I/O control signals was easy, but some of the others are not so obvious. The $\overline{INTA}$ (interrupt acknowledge) signal is derived by ANDing the $\overline{IORQ}$ (I/O request) and $\overline{M1}$ (first machine cycle) outputs. The H8 uses an M1 signal and decodes it by ANDing the 8080's $PD_5$ and SYNC outputs. Since the Z-80 already has an $\overline{M1}$ output, I simply ran it (through an inverter) to the SYNC input of the 8224 and tied $PD_5$ (actually, $D_5$ on the 8238) high through a resistor.

The $\overline{WAIT}$ input on the Z-80 is the same as the READY input on the 8080, except that it does not have to be timed, so I connected the raw RDYIN input at the 8224 to the Z-80 $\overline{WAIT}$ input. The Z-80 $\overline{INT}$, $\overline{BUSRQ}$ and $\overline{BUSAK}$ signals are just inverted counterparts of the 8080 INT, HOLD and HLDA pins, so I connected them together through inverters. The $\overline{NMI}$ (non-maskable interrupt) on the Z-80 is not used, so I tied it high. (Later on, I may write a monitor that uses that interrupt to return to monitor control from a user program, which would make possible debugging a program that had interrupts disabled.)

The $\overline{HALT}$ and $\overline{RFSH}$ outputs of the Z-80 are not needed, and were left unconnected. The address pins on the Z-80 were connected to the corresponding pins on the 8080 socket, and the data pins go to the DB pins on the 8238 socket.

The Z-80 does not produce a counterpart to the 8080 INTE (interrupt enable flip-flop) output. The H8 uses that signal to light an LED on the front panel and to operate the single instruction (SI) button. INTE is produced by the optional section of the circuit, and non-machine-language hackers who never use the SI button may leave that section out and tie pin 16 on the 8080 socket high instead. If you do that, the SI button will do nothing, and the ION light will always be on.

The INTE output of the 8080 is set or reset according to whether interrupts are enabled or disabled. Interrupts may be enabled and disabled by software, using the EI and DI instructions, and are always disabled when the processor receives an interrupt. The HZ8 adapter circuitry responds only to EI and DI, but that is sufficient to achieve normal operation of the H8. It does this by examining the data bus at M1 time, when op codes are fetched.

The EI instruction in binary is 11111011, and DI is 11110011. Bit 3, the only one that changes, goes to the data input of a D-type flip-flop, while the others, along with M1, are used to clock the data through when they are all present. Capacitor $C_1$ slows the clocking down just enough to ensure that $D_3$ has settled down before it is sampled.

## Single Stepping

Before I designed the INTE circuitry on the adapter, I looked into various software single-step schemes, which I considered using, and found that they are all deficient in some way. Most cannot handle certain instructions, and all require considerably more code than the Heath method. The H8 can single-step through all of the 8080's instructions, and, with my adapter, through all of the Z-80's codes in only a few bytes of

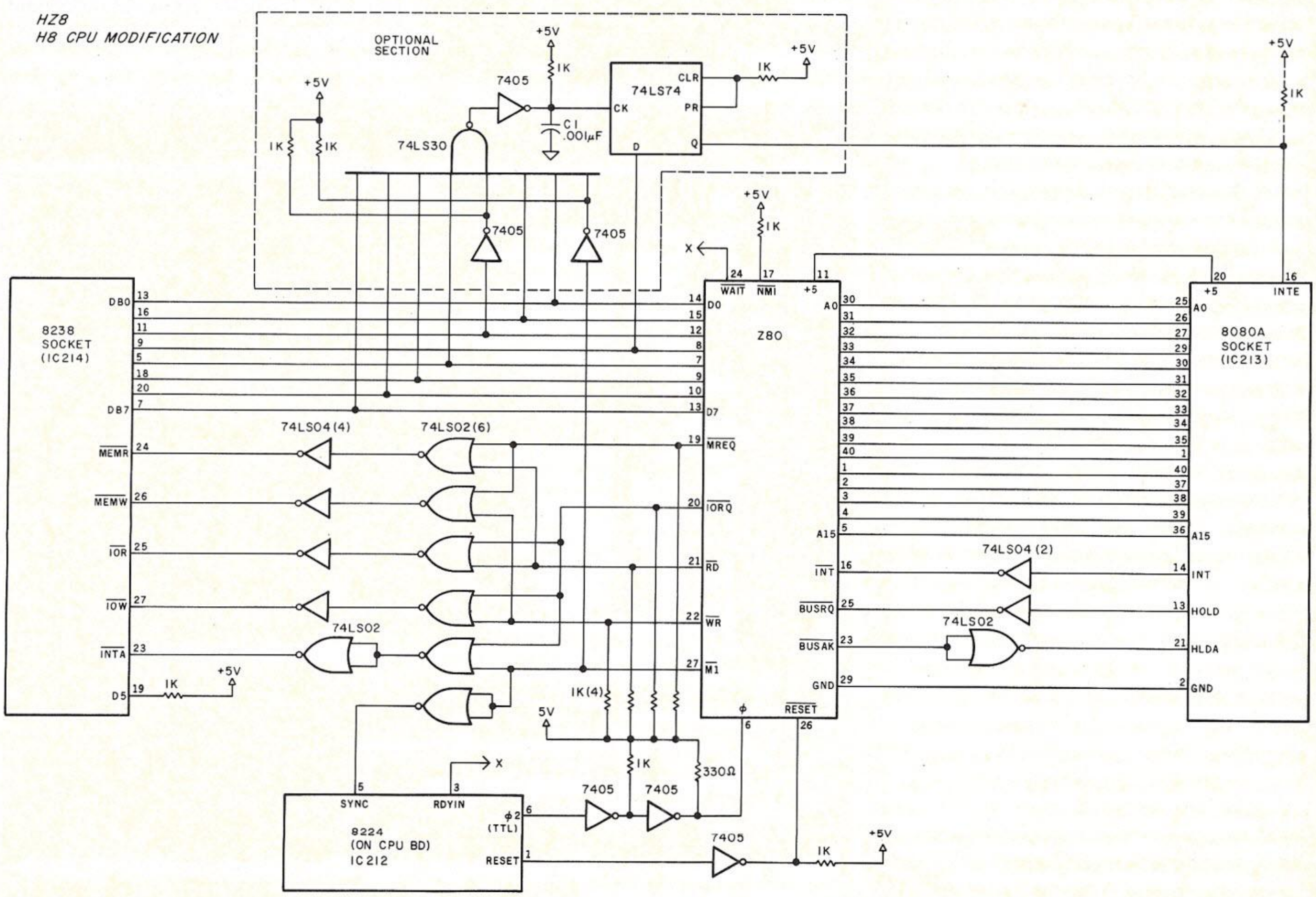



*Fig. 1. Z-80 adapter circuit. Connect the points marked "X"; don't tie INTE (pin 16) on the 8080 socket high if optional section isn't built.*

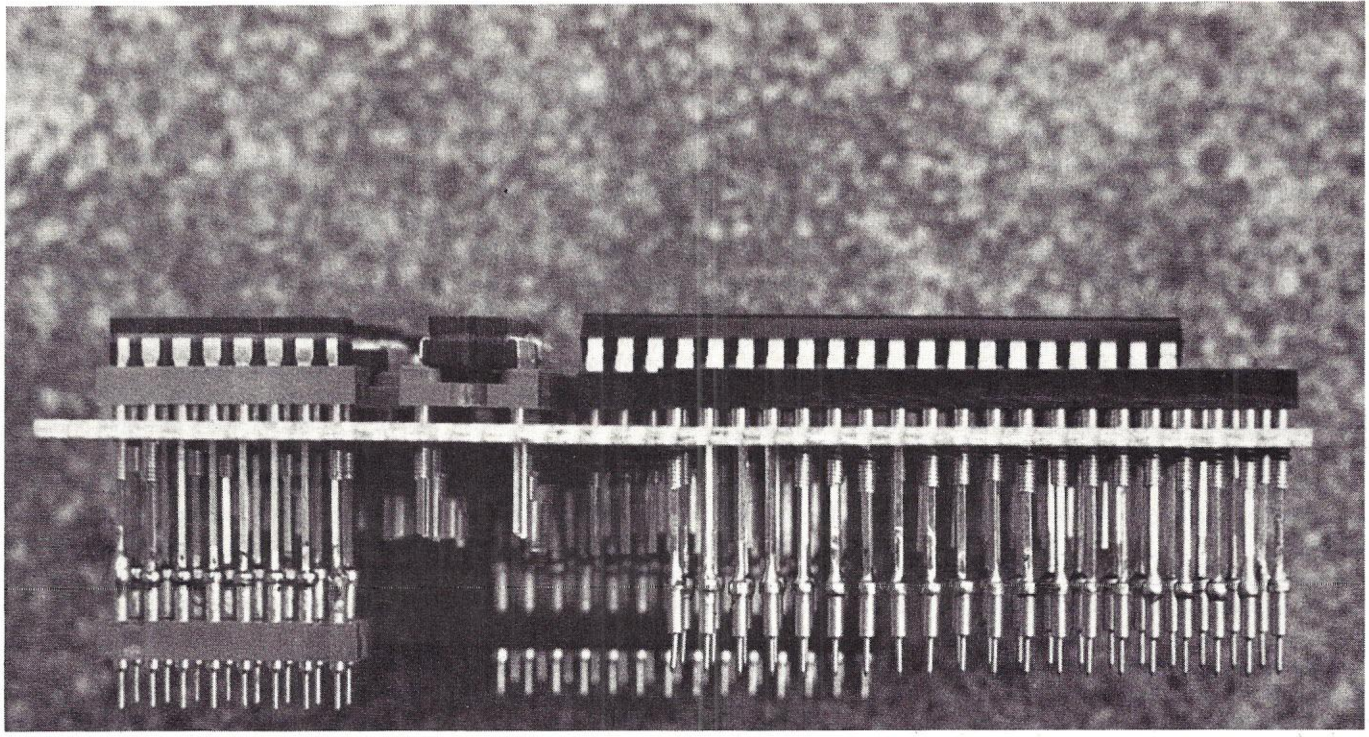


*Photo 2. Side view of adapter board shows the frameless socket pins attached to the ends of the wire-wrap pins, using an ordinary framed socket (left) for alignment. If pins remain at full length, the CPU card won't fit in the first motherboard position.*

code.

When you press the SI on the H8 front panel, the monitor jumps to a routine that first disables interrupts, bringing the INTE output low. Then data is written to a port which causes the output of a flip-flop to go high. This signal goes to a NAND gate, the output of which will produce a level 2 interrupt when its other input goes high. That input is controlled by two flip-flops whose data is the INTE signal, with M1 used as the clock. It takes two M1s to clock the current value of INTE to the NAND gate.

After writing to the port mentioned above, the software restores all user registers and flags (previously saved when entering the monitor mode). It then enables interrupts, bringing INTE high, and jumps to the user program. That jump sends out one M1 pulse, and the first user instruction sends out another, allowing the INTE signal to generate an interrupt. The processor allows the current instruction to finish, and then control is returned to the monitor. In this way, one user program instruction is executed each time you press the SI button.

The single phase system clock required by the Z-80 is a special case. It could be supplied by the Ø2 (TTL) output of the 8224, except that it requires a greater voltage swing than TTL. The signal must go from a low of no more than 0.8 volts to a high of no less than 4.4 volts (with a 5 volt supply). I solved this problem by running the clock through two gates of a 7405 open collector inverter, with pull-up resistors on the outputs.

To ensure a fast rise-time to the higher-than-normal voltage, a 330-ohm resistor pulls up the input to the Z-80. Another gate of the 7405 is used to invert the 8224 RESET to supply the Z-80's inverted version of the signal, and the rest are used in the optional section.

### Construction

Using the wire-wrap technique, I built the adapter on a 3 × 4 inch piece of standard perfboard with 0.1 inch spaced holes. I placed 40-pin, 28-pin and 16-pin wire-wrap sockets on the boards so that their pins are directly over the holes in the 8080, 8238 and 8224 sockets in the CPU board. The Heath-supplied X-ray view of the PC board can be used to align the sockets. Another 40-pin

**ICs**

| Qty. | Type | Function |
|---|---|---|
| 2 | 74LS02 | Quad 2-Input NOR |
| 1 | 74LS04 | Hex Inverter |
| 1 | 74LS30 | 8-Input NAND* |
| 1 | 74LS74 | Dual D-type Flip-Flop* |
| 1 | 7405 | Hex Open Collector Inverter |
| 1 | Z80 | Microprocessor |

*Optional

**Sockets**

| Qty. | Pins | Augat Part No. |
|---|---|---|
| 6 (4) | 14 WW | 514-AG10F |
| 2 | 16 WW | 516-AG10F |
| 1 | 28 WW | 528-AG10F |
| 2 | 40 WW | 540-AG10F |
| 1 | 16 F | 716-AG4D |
| 1 | 28 F | 728-AG4D |
| 1 | 40 F | 740-AG4D |

WW = Wire Wrap  F = Frameless

16-, 28-, and 40-pin standard or wire-wrap sockets are also temporarily required to mount frameless socket pins.

**Miscellaneous**

1 Dale No. LDP16-02-102G resistor pack or equivalent
or
12 1k ohm resistors and 1 330 ohm resistor
3 × 4 inch perfboard
Wire

*Table 1. HZ8 parts list. I built the prototype using 16-pin sockets for 14-pin ICs due to availability (or lack thereof).*

wire-wrap socket, which holds the Z-80, was installed between the other 40-pin and 28-pin sockets. Placement of the other sockets is not important, but wire lengths should be kept short.

Power and ground for the TTL were derived from the VCC and ground pins of the 8238 socket, except for the 7405, which gets its ground and power from the 8224 socket. The two extra chips for the optional section were added later, and get their power from the Z-80 socket. For the pull-up resistors, I used a resistor package containing 15 resistors, all 1k ohm, connected internally to a common pin. To make the 330 ohm resistor, three resistors in the package were paralleled. Discrete resistors may also be used.

After being wired, the circuit should be checked with an ohmmeter or continuity tester for correctness, because when it is finished it will be difficult to make changes. To make it possible to plug the adapter into the CPU board, frameless socket pins were soldered to the ends of the wire-wrap pins on the 8080, 8238 and 8224 sockets. Frameless sockets are socket pins that are installed on an aluminum frame that is removed after the sockets are soldered into a PC board.

To connect these pins to a wire-wrap socket, they should first be removed from the frame and plugged into an ordinary framed socket. That will hold them in place while they are soldered to the wire-wrap pins.

Photo 2 shows how the pins are connected. If the wire-wrap pins are left their full length, the CPU card with the adapter installed will not fit in the first motherboard position. If the wire-wrapping is kept close to the board, and all pins are cut so that they protrude no more than 1/4 inch from the perfboard (before the frameless pins are attached), then the board will just fit in the first slot.

Those H8ers capable of making their own PC boards and using that technique would not have a thickness problem. The frameless socket pins could be soldered directly to the back of the PC board over the protruding pins of the sockets.

### Installation and Checkout

The adapter cannot be plugged into the CPU board unless the 8080, 8238 and 8224 sockets on the board are all of the flat, low-profile type. If any are not, they will have to be changed. The 8080 and 8238 ICs should be stored in anti-static material (foil will do) while they are not in use.

After the TTL ICs, resistor pack and 8224 are installed, the adapter can be plugged into the CPU board. If the CPU is to be used in the second motherboard slot, framed sockets can be plugged onto the socket pins to protect them before the adapter board is plugged in. Then the Z-80 can be installed, and the CPU board replaced in the computer.

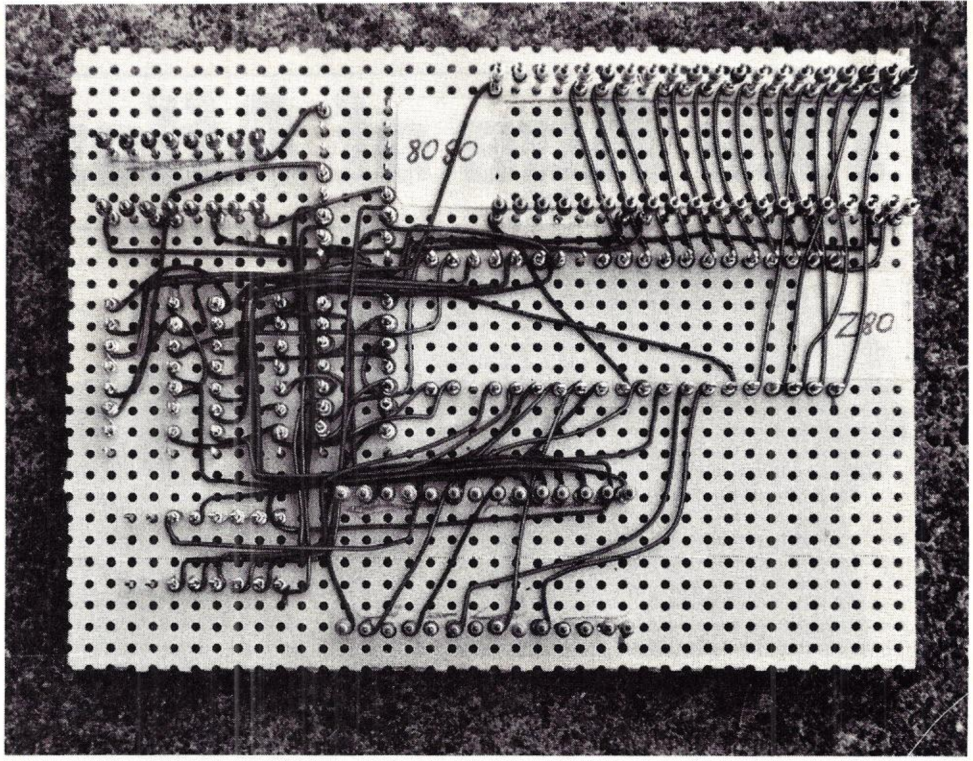



*Photo 3. Bottom view of the adapter showing wire-wrap technique.*

Checkout is simply a matter of turning the computer on. If PAM-8 (the front-panel monitor) signs on normally by lighting the displays and beeping the horn, everything else should also run normally. If there is trouble, the first area to check is the wiring and then the chips.

### The Z-80

The 8080 instruction set consists of 244 individual op codes. Like most 8-bit processors, each op code consists of one byte, making 256 the total number of codes possible. The Z-80, however, uses four of the 12 op codes not used by the 8080 as the first byte of several two-byte op codes. You can think of these as 16-bit op codes that are fetched one byte at a time. In that sense, the Z-80 is a predecessor of the Intel 8088, a 16-bit micro designed to use an 8-bit data bus.

The two-byte op codes are used for a variety of purposes, including 16-bit arithmetic, bit manipulation, working with two index registers and some versatile block move and search instructions. These last types are an elementary form of microcode, another way in which the Z-80 looks forward to the big machines.

Two of the Z-80's 16-bit instructions adversely affect the operation of the optional section of the adapter circuit during normal running (but not during single stepping). The second bytes of these instructions, SET 6,E and SET 7,E, have the same binary code as DI and EI. Since the Z-80 issues an M1 pulse for each byte of two-byte op codes, the adapter circuitry sees the second bytes of those SET instructions as DI and EI. The result is that the front panel ION light may be lying if those instructions are in the code. Upon return to monitor, however, an EI instruction is encountered, and proper indication is restored.

The Z-80 uses five of the other eight unused 8080 op codes for an unconditional and four conditional relative jumps. In the 8080, all jumps require three bytes—one for the op code and two for the address. With the Z-80 relative jumps, a single byte following the op code specifies the jump destination as a signed 8-bit offset added to the program counter.

Another of the unused 8080 codes is a special relative jump instruction that decrements the B register each time it is executed and jumps if B is greater than zero. The remaining two extra op codes are used to switch between alternate sets of the six general-purpose registers and alternate flag registers and accumulators. In all, the Z-80 has about 700 op codes in its instruction set.

### A Final Note

Those who use Heath's cassette assembler, HASL-8 version 4.01.01 or 4.02.00, will have to make a patch before it will work on a Z-80. The program contains one of the 8080's unused op codes, 40 (octal), which is one of the Z-80's relative jumps. To correct this bug in version 4.01.01, change the contents of address 055.265 (split octal) to 000. To correct version 4.02.00, change 055.365 to 000. ■

# The Age of Affordable Pers

In 1978 Ohio Scientific introduced a revolutionary new low cost computer — the Superboard II. This computer provides all important personal computer features on a single board at a cost of under $300. The Superboard II received rave reviews by microcomputer experts such as:

**"We can heartily recommend the Superboard II computer system for the beginner who wants to get into microcomputers with a minimum of cost. Moreover, this is a 'real' computer with full expandability."**
POPULAR ELECTRONICS MARCH, 1979

**"The Superboard II weighs in at $279 and provides a remarkable amount of computing for this incredible price."**
KILOBAUD MICROCOMPUTING FEBRUARY, 1979

**"The Superboard II and its fully dressed companion the Challenger 1P series incorporate all the fundamental necessities of a personal computer at a very attractive price. With the expansion capabilities provided, this series becomes a very formidable competitor in the home computer area."**
INTERFACE AGE APRIL, 1979

**"The graphics available permit some really dramatic effects and are relatively simple to program . . . The fact that the system can be easily expanded to include a floppy means that while you are starting out with a low-cost minimal system, you don't have to throw it away when you are ready to go on to more complex computer functions. At $279, Superboard II is a tough act to follow."**
RADIO ELECTRONICS JUNE, 1979

**"The Superboard is an excellent choice for the personal computer enthusiast on a budget."**
BYTE MAY, 1979

Since the introduction of Superboard II, the cost of personal computers has actually gone up with new models by major manufacturers ranging from $1000 to well over $4000 due to the general cost of inflation and the increasing functionality included in these computers. Today Cleveland Consumer Computers is offering you the original Superboard II at its original price of just $279. In today's economy this is by far the best buy in personal computing ever!

**The Superboard II** can entertain your whole family with spectacular video games and cartoons, made possible by its ultra high resolution graphics and super fast BASIC. It can help you with your personal finances and budget planning, made possible by its decimal arithmetic ability and cassette data storage capabilities. It can assist you in school or industry as an ultra powerful scientific calculator, made possible by its advanced scientific math functions and built-in "immediate" mode which allows complex problem solving without programming! This computer can actually entertain your children while it educates them in topics ranging from naming the Presidents of the United States to tutoring trigonometry — all possible by its fast extended BASIC, graphics and data storage ability.

The machine can be economically expanded to assist in your business, remotely control your home, communicate with other computers and perform many other tasks via the broadest line of expansion accessories in the microcomputer industry.

This machine is super easy to use because it communicates naturally in BASIC, an English-like programming language. So you can easily instruct it or program it to do whatever you want, but you don't have to. You don't because it comes with a complete software library on cassette including programs for each application stated above. Ohio Scientific also offers you hundreds of inexpensive programs on ready-to-run cassettes. Program it yourself or just enjoy it; the choice is yours.

The Superboard II comes fully assembled and tested. It requires +5V at 3 Amps and a video monitor or TV with RF converter to be up and running. $279.00

## Standard Features:

- Uses the ultra powerful 6502 Microprocessor.
- 8K Microsoft BASIC-in-ROM. Full feature BASIC runs faster than currently available personal computers and all 8080 based business computers.
- 4K static RAM on board expandable to 8K.
- Full 53-key keyboard with upper/lower case and user programmability.
- Kansas City standard audio cassette interface for high reliability.
- Full machine code monitor and I/O utilities in ROM.

# onal Computing is Still Here.

Mini-Floppy Disk Drive

Direct access video display has 1K of dedicated memory (besides 4K user memory), features upper case, lower case, graphics and gaming characters for an effective screen resolution of up to 256 x 256 points. Normal TV's with overscan display about 24 rows of 24 characters without overscan up to 30 x 30 characters.

## Optional Extras:

- Available 610 expander board features up to 24K static RAM (additional), dual mini-floppy interface, and an OSI 48 line expansion interface.
- Assembler/Editor and Extended Machine Code monitor available.
- 630 I/O Expander. RGB color and NTSC composite color outputs with up to 16 colors, Dual 8-axis joystick interface, AC remote control interface which mates with AC-12P, home security interface which mates with the AC-17P, 16-line parallel I/O interface, 16-pin I/O bus interface which allows the connection of parallel I/O lines or high speed analog I/O module, or a PROM blaster or solderless interface prototyping board, programmable sound generator and program selectable modem and high speed printer ports, and more.

**Freight Policies** All orders of $100 or more are shipped freight prepaid. Orders of less than $100 please add $4.00 to cover shipping costs. Ohio Residents add 5.5% Sales Tax.

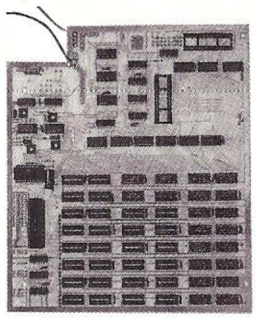
**Guaranteed Shipment** Cleveland Consumer Computers & Components guarantees shipment of computer systems within 48 hours upon receipt of your order. **Our failure to ship within 48 hours entitles you to $35 of software, FREE.**

**Hours:**
Call Monday thru Friday
8:00 AM to 5:00 PM E.D.T.

## Software:

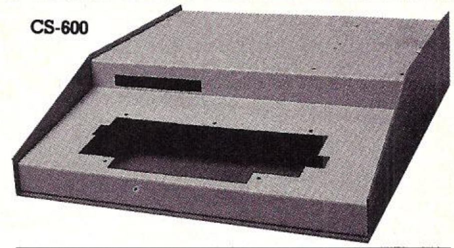
Ohio Scientific and independent suppliers offer hundreds of programs for the Superboard II, in cassette and mini-floppy form. Here is a sampling of popular Ohio Scientific programs for the Superboard II.

| EDUCATIONAL PROGRAMS | SBII&C1P | Price |
|---|---|---|
| BASIC Tutor Series | SCE-336 | $35.00 |
| Clock Tutor | SCE-353 | 6.50 |
| Continents Quiz | SCE-332 | 6.50 |
| Definite Integral | SCE-326 | 6.50 |
| French Drill & Tutor | SCE-339 | 6.50 |
| German Tutor & Drill | SCE-342 | 6.50 |
| Hangman (8K) | SCE-324 | 9.00 |
| Log Tutors 1-3 | SCE-344 | 6.50 |
| Math Blitz | SCE-329 | 6.50 |
| Math Intro | SCE-319 | 6.50 |
| Mathink | SCE-337 | 9.00 |
| Matrix Tutors 1-3 | SCE-345 | 6.50 |
| Metric Tutor & Quiz | SCE-335 | 6.50 |
| Spanish Drill & Tutor | SCE-352 | 6.50 |
| Spelling Quiz | SCE-333 | 6.50 |
| Trig Tutor (8K) I & II | SCE-318 | 6.50 |
| **BUSINESS PROGRAMS** | | |
| Address Book | SCB-523 | 9.00 |
| Advertisement Demo | SCB-520 | 6.50 |
| Inventory Demo | SCB-518 | 6.50 |
| Mailing List (8K) | SCB-524 | 6.50 |
| Straight & Constant Depreciation | SCB-500 | 9.00 |
| Time Calculator | SCB-525 | 9.00 |
| **PERSONAL PROGRAMS** | | |
| Biorhythm | SCP-716 | 9.00 |
| Calorie Counter | SCP-708 | 6.50 |
| Checking Account | SCP-719 | 9.00 |
| Loan Finance | SCP-717 | 6.50 |
| Personal Calendar | SCP-718 | 6.50 |
| Savings Account | SCP-720 | 9.00 |
| **GAME PROGRAMS** | | |
| Baseball I | SCG-975 | 6.50 |
| Black Jack | SCG-955 | 6.50 |
| Civil War | SCG-977 | 6.50 |
| Destroyer | SCG-951 | 6.50 |
| High Noon | SCG-960 | 6.50 |
| Hockey | SCG-979 | 6.50 |
| Lander | SCG-925 | 6.50 |
| New York Taxi | SCG-956 | 6.50 |
| Poker | SCG-962 | 6.50 |
| Racer | SCG-949 | 6.50 |
| Space War | SCG-942 | 6.50 |
| Star Trek | SCG-946 | 6.50 |
| Star Wars | SCG-926 | 6.50 |
| Tic-Tac-Toe | SCG-945 | 6.50 |
| Tiger Tank | SCG-950 | 14.00 |



610 Board

CS-600



## Hardware:

| | | |
|---|---|---|
| **Superboard II** | as specified in the advertisement. | **$279** |
| **610 Board** | For use with Superboard II and Challenger 1P, 8K static RAM expandable to 24K or 32K system total. Accepts up to two mini-floppy disk drives. Requires +5V@ 4.5 amps. | **298** |
| **Mini-Floppy Disk Drive** | Includes Ohio Scientific's PICO DOS software and connector cable. Compatible with 610 expander board. Requires +12V@ 1.5 amps and +5V@ 0.7 amps. | **299** |
| **630 Board** | As specified in the advertisement. | **229** |
| **AC-3P** | 12" combination black and white TV/video monitor. | **159** |
| **4KP** | 4K RAM chip set. | **79** |
| **PS-005** | 5V 4.5 amp power supply for Superboard II. | **35** |
| **PS-003** | Mini-floppy power supply. | **29** |
| **C1P Sams** | C1P/Superboard II Manual. | **8** |
| **OS-65D** | V3.2 Disk Operating System with 9-digit extended BASIC, random access and sequential files. | **49** |
| **CS-600** | Metal case for Superboard II, 610 and 630 board and two power supplies. | **49** |
| **CS-610** | Metal case for single floppy disk drive and power supply. | **49** |
| **AC-12P** | Wireless AC remote control system. Includes control console, two lamp modules and two appliance modules for use with 630 board. | **175** |
| **AC-17P** | Home security system. Includes console, fire detector, window protection devices and door unit for use with 630 board. | **249** |
| **C4P Sams** | C4P Manual. | **16** |
| **C3 Sams** | Challenger III Manual. | **40** |

**To Order:**
Or to get our free catalog **CALL 1-800-321-5805 TOLL FREE.**
Charge your order to your **VISA** or **MASTER CHARGE** ACCOUNT
Ohio Residents Call: (216) 464-8047. Or write, including your check or money order, to the address listed below.

**CLEVELAND CONSUMER COMPUTERS & COMPONENTS** ✓28
**P.O. Box 46627**
**Cleveland, Ohio 44146**

## Order Form: CLEVELAND CONSUMER COMPUTERS & COMPONENTS P.O. Box 46627 Cleveland, Ohio 44146

☐ Superboard II $279. ☐ 630 Board $229.
☐ 610 Board $298. ☐ AC-3P 12" B-W Monitor $159.
☐ Mini-Floppy Disk Drive $299. ☐ C1P Sams Manual $8.
(Attach separate sheet for other items.)

NAME:____________________
ADDRESS:____________________
CITY:__________ STATE:__________ ZIP:__________
Payment by: VISA: __ MASTER CHARGE: __ MONEY ORDER: __
Credit Card Account #____________________
Expires:__________ Interbank # (Master Charge)__________
TOTAL CHARGED OR ENCLOSED: $________ (Ohio Residents add 5.5% Sales Tax)
All orders shipped insured UPS unless otherwise requested. FOB Cleveland, Ohio.

*Robert M. Richardson*
*Drawer 1065*
*Chautauqua Lake, NY 14722*

# Level II ROM Subroutine Test

*Talk to your TRS-80 in its own language.*

| Function | Number Type | Decimal | Hexadecimal |
|---|---|---|---|
| ABS | 2-4-8 | 2423 | 0977 |
| ATN | 4-8 | 5565 | 15BD |
| BASIC | (Return L-II) | 6681 | 1A19 |
| BASIC | (Return disk) | 112 | 0075 |
| BREAK | (RST address) | 16396 | 400C |
| CDBL | 2-4 | 2779 | 0ADB |
| CINT | 4-8 | 2687 | 0A7F |
| CLS | 2-4-8 | 457 | 01C9 |
| COS | 4-8 | 5441 | 1541 |
| CSNG | 2-8 | 2737 | 0AB1 |
| EXP | 4-8 | 5177 | 1439 |
| FIX | 2-4 | 2854 | 0B26 |
| INT | 2 | 2871 | 0B37 |
| INVERT SIGN | 2 | 3153 | 0C51 |
| INVERT SIGN | 4-8 | 2434 | 0982 |
| LOG | 4-8 | 2057 | 0809 |
| MEMORY | (Size input) | 181 | 00B5 |
| RANDOM | 2-4-8 | 467 | 01D3 |
| RETURN | (To subroutine) | 32000 | 7D00 |
| RND <1.0 | 4-8 | 5321 | 14C9 |
| SGN | 2 | 2442 | 098A |
| SIN | 4-8 | 5447 | 1547 |
| SQR | 4-8 | 5095 | 13E7 |
| TAN | 4-8 | 5544 | 15A8 |

*Table 1. Level II arithmetic/trigonometric conversion table. Number types: 2 = integer; 4 = single precision; 8 = double precision.*

*Listing 1. Source code.*

```
W4UCH   EQU     7D00H           ;7D00H = 32000 DECIMAL
        ORG     W4UCH           ;PROGRAM WILL START HERE
        LD      A,4EH           ;4EH="N"=NUMBER DESIRED ?
        CALL    032AH           ;DISPLAY "N" ON VIDEO
        CALL    1BB3H           ;KYBD/VIDEO INPUT ROUTINE
        RST     10H             ;SCAN STRING - SET C FLAG
        CALL    0E6CH           ;ASCII-ACCUM RET MINIMUM
RETURN  EX      AF,AF'          ;EXCHANGE REGISTERS-
```

Here is another interesting test program for the advanced assembly-language programmer. It lets you access and test many of the arithmetic/trigonometric subroutines in the TRS-80 Level II ROM written by Microsoft's Bill Gates and Paul Allen.

The beginning assembly-language programmer should certainly learn how to write fundamental arithmetic/trig functions by himself, but once these techniques have been mastered as part of the learning process, it is inefficient and unnecessary to duplicate in assembly language those subroutines already in the Level II ROM.

Table 1 lists those functions and their addresses that may be accessed and tested by this mini-program that only occupies 144 bytes of high memory and may be entered using the TRS-80 Editor/Assembler in about five minutes.

Listing 1 is a printout of the test program's source code, and Listing 2 shows the program's object code. As you will see, the majority of this program is written using Level II ROM subroutines. Were these subroutines not used in this particular assembly-language test program, it would require ten times as much program memory and occupy 550 rather than 55 assembly-language program lines.

### Program Flow

The comments included with the source code program are largely self-explanatory

and delineate each line's function. This program operates equally well with non-disk Level II, DOS 2.1, DOS 2.2 and NEWDOS +. Program operation is as follows:

1. Load the program under the SYSTEM or DOS command. Give it any name you wish. We like the program name DISCOV, for discovery, since that is what the program is all about. After loading is complete, type in /32000 to activate the program (with disk you must first load BASIC, then type SYSTEM, ENTER, and then type in /32000 ENTER, if you load the program in DOS).

2. The letter 'N?' will appear on the video. The program is asking you for a number to work on. Enter any number up to 16 digits, depending on the function you wish to test. Let us start out with a simple example by entering the number 100000.

3. The numbers '2' '100000' will appear on the next line of the video display. The '2' is the number type brilliantly calculated by the Level II ROM. Since we are dealing only with numbers in this article, we will skip over strings et al for the time being. The number types are as follows: 2 = integer, 4 = single precision and 8 = double precision. Table 1 lists those operations that can be performed on a number for a given number type; e.g., it is against the rules to take the square root SQR of an integer. We must first change it.

4. On the following line of the video display you will see 'C?'. The program is asking you what type of conversion you wish. Let's enter 2737, which is the address of the CSNG function, to change our number from an integer to single precision, then ENTER. The next line will show '4' '1000'. We now have a single-precision number to work with, so let's try taking its square root by typing in 5095, the address of the SQR routine, then ENTER. The next line shows '100'. This sure is easier than writing a complete stand-alone assembly-language square root subroutine. Let's try it again. Type in 5095 ENTER. Again, the line below displays the square root, this time, the numeral 10.

5. To insert a new number to try your program on, merely type in 32000 ENTER. This brings us back to where we started by displaying 'N?'. Thus, 32000 is our subroutine. Our assembly-language program does not discriminate between ROM or RAM; it could care less.

6. We could go on and on converting numbers such as deriving the natural LOG of any number and then restoring it to its original value via the EXP function, and/or deriving the TANgent of a number, then its arc tangent ATN and then the TANgent again . . . ad infinitum. You may escape this conversion routine any time you wish by typing 6681 ENTER, which will take you back to BASIC with a READY displayed. To return to your conversion routine, type SYS-

```
            EXX                     ;TO PRESERVE VALUES.
            LD      DE,411DH        ;MOVE MEM ACCUM DATA FROM
            LD      HL,STORE        ;TO TEMPORARY STASH.
            LD      B,8             ;NUMBER OF BYTES TO MOVE
            CALL    09D7H           ;MOVE IT - SUBROUTINE
            LD      DE,4127H        ;MOVE CDBL DATA FROM-
            LD      HL,CDBL         ;TO TEMPORARY STASH.
            LD      B,8             ;NUMBER OF BYTES TO MOVE
            CALL    09D7H           ;MOVE IT - SUBROUTINE
            LD      A,(40AFH)       ;NUMBER TYPE MEM LOCATION
            LD      (FLAG),A        ;MOVE TO TEMPORARY STASH
            ADD     A,48            ;CONVERT TO ASCII NUMBER
            CALL    032AH           ;DISPLAY NUMBER TYPE
            LD      A,20H           ;20H = ASCII SPACE
            CALL    032AH           ;DISPLAY SPACE ON VIDEO
            CALL    0FBDH           ;CONV MEM ACCUM TO ASCII$
            CALL    28A7H           ;DISPLAY CONVERTED NUMBER
            LD      A,0DH           ;0DH=SKIP A LINE/CARR RTN
            CALL    032H            ;DO IT - ON VIDEO DISPLAY
            LD      A,43H           ;"C" = CONVERSION NUMBER?
            CALL    32AH            ;DISPLAY "C" ON VIDEO
            CALL    1BB3H           ;KYBD/VIDEO INPUT ROUTINE
            RST     10H             ;SCAN STRING - SET C FLAG
            CALL    0E6CH           ;ASCII-ACCUM RET MINIMUM
            CALL    0A7FH           ;CONVERT TO INTEGER
            LD      (CONV),HL       ;STORE CONVERSION ADDRESS
            LD      DE,CDBL         ;MOVE CDBL DATA FM STASH-
            LD      HL,4127H        ;TO PERMANENT ADDRESS.
            LD      B,8             ;NUMBER OF BYTES TO MOVE
            CALL    09D7H           ;MOVE IT - SUBROUTINE
            LD      DE,STORE        ;MOVE MEM ACCUM FM STASH-
            LD      HL,411DH        ;TO PERMANENT ADDRESS.
            LD      B,8             ;NUMBER OF BYTES TO MOVE
            CALL    09D7H           ;MOVE IT - SUBROUTINE
            LD      A,(FLAG)        ;NUMBER TYPE FROM STASH-
            LD      (40AFH),A       ;TO PERMANENT ADDRESS.
            LD      HL,RETURN       ;RETURN MEM LOCATION-
            PUSH    HL              ;LOADED INTO STACK.
            LD      HL,(CONV)       ;CONVERSION MEM LOCATION-
            PUSH    HL              ;LOAD ON TOP OF STACK.
            EX      AF,AF'          ;RESTORE REGISTERS-
            EXX                     ;TO ORIGINAL VALUES.
            RET                     ;SNEAKY CALL-TOP OF STACK
FLAG        DEFS    1               ;NUMBER TYPE STASH
CONV        DEFS    2               ;CONVERSION ADDRESS STASH
CDBL        DEFS    8               ;CDBL DATA STASH
STORE       DEFS    8               ;ACCUMULATOR STASH
            END     W4UCH           ;AMATEUR RADIO CALL LTRS
```

*Listing 2. Object code.*

```
7D00                W4UCH   EQU     7D00H
7D00                        ORG     W4UCH
7D00  3E4E                  LD      A,4EH
7D02  CD2A03                CALL    032AH
7D05  CDB31B                CALL    1BB3H
7D08  D7                    RST     10H
7D09  CD6C0E                CALL    0E6CH
7D0C  08            RETURN  EX      AF,AF'
7D0D  D9                    EXX
7D0E  111D41                LD      DE,411DH
7D11  21837D                LD      HL,STORE
7D14  0608                  LD      B,8
7D16  CDD709                CALL    09D7H
7D19  112741                LD      DE,4127H
7D1C  217B7D                LD      HL,CDBL
7D1F  0608                  LD      B,8
7D21  CDD709                CALL    09D7H
7D24  3AAF40                LD      A,(40AFH)
7D27  32787D                LD      (FLAG),A
7D2A  C630                  ADD     A,48
7D2C  CD2A03                CALL    032AH
7D2F  3E20                  LD      A,20H
7D31  CD2A03                CALL    032AH
7D34  CDBD0F                CALL    0FBDH
7D37  CDA728                CALL    28A7H
7D3A  3E0D                  LD      A,0DH
7D3C  CD3200                CALL    032H
7D3F  3E43                  LD      A,43H
7D41  CD2A03                CALL    32AH
7D44  CDB31B                CALL    1BB3H
7D47  D7                    RST     10H
7D48  CD6C0E                CALL    0E6CH
```

```
7D4B CD7F0A      00420         CALL    0A7FH
7D4E 22797D      00430         LD      (CONV),HL
7D51 117B7D      00440         LD      DE,CDBL
7D54 212741      00450         LD      HL,4127H
7D57 0608        00460         LD      B,8
7D59 CDD709      00470         CALL    09D7H
7D5C 11837D      00480         LD      DE,STORE
7D5F 211D41      00490         LD      HL,411DH
7D62 0608        00500         LD      B,8
7D64 CDD709      00510         CALL    09D7H
7D67 3A787D      00520         LD      A,(FLAG)
7D6A 32AF40      00530         LD      (40AFH),A
7D6D 210C7D      00540         LD      HL,RETURN
7D70 E5          00550         PUSH    HL
7D71 2A797D      00560         LD      HL,(CONV)
7D74 E5          00570         PUSH    HL
7D75 08          00580         EX      AF,AF'
7D76 D9          00590         EXX
7D77 C9          00600         RET
0001             00610 FLAG    DEFS    1
0002             00620 CONV    DEFS    2
0008             00630 CDBL    DEFS    8
0008             00640 STORE   DEFS    8
7D00             00650         END     W4UCH
00000 TOTAL ERRORS
```

TEM, then ENTER and type /32000, then ENTER.

### Conclusion

This article covers only a few of the subroutines in Level II BASIC ROM. Assembly-language programming is the ne plus ultra of serious computing. Your assembly-language program runs 300 times faster than the same program in BASIC and uses only 1/10th as much memory.

Learning to talk to your computer in its own language rather than through an interpreter (BASIC, FORTRAN or Pascal) is a most satisfying and rewarding experience if you have the patience and fortitude to master it. ■

The material in this article comes from *The Disassembled Handbook for TRS-80*, which is available from the author for $10 postpaid.

# KILOBAUD KLASSROOM NO. 21

# Expansions and Programming

*Now that your computer is built, you'll want to expand it for more memory and I/O capability. . . and then on to programming.*

*Peter A. Stark*
*PO Box 209*
*Mt. Kisco, NY 10549*

I've described how to build the basic Kilobaud Klassroom Komputer. This time I'll show some ways in which the computer can be expanded and begin my discussion of computer programming.

Though we did not originally intend our computer to be anything but a small control computer, a number of readers have asked for expansion information so they could make it into a more general-purpose system.

With some of the expansions described below, it is possible to run some programs that people associate only with larger systems. (For example, a machine-language monitor for entering, executing and debugging machine-language programs, as well as a Tiny BASIC, are available from Star-Kits, PO Box 209, Mt. Kisco NY 10549.)

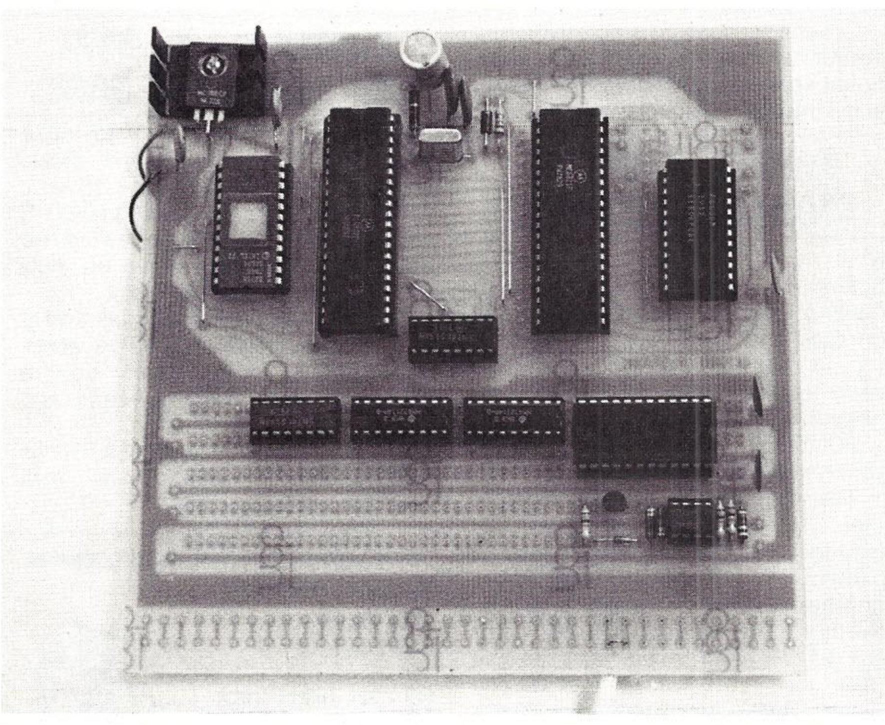


*Photo 1. This unit has been expanded with both the RS-232C serial port as well as an additional 1K of RAM.*

### Installing a Second PIA

The printed circuit board layout I presented last time has one socket that is fully wired for a PIA parallel interface chip, and a second socket that can be used either for a second 6820 (or 6821) PIA or for a 6850 ACIA for serial communications.

This socket is located in the top right corner of the PC board (see Photo 1). Since the PIA and ACIA connections are slightly different, only some of the pins of this socket position are already connected (the data bus and clock connections for the two ICs are in the same relative positions). The other connections, however (those that are different for the two ICs), must be wired by hand for the IC you plan to use.

Installing a PIA is easy. Just wire the following pins:

- Pin 24, CS1, connects to +5 volts. Pin 24 connects to a small round pad right next to it; just next to that is another round pad, which is at +5 volts. Connect a short jumper between them.
- Pin 34, $\overline{\text{RESET}}$, connects to pin 34 of the other PIA. Small round pads are located next to pin 34 of both PIAs to make the connection easier.
- Pin 35, RS1, connects to pin 35 of the other PIA. Round pads located next to pin 35 of both PIAs are used.

In addition, both PIAs have small pads next to pins 2 through 19 and pins 39 and 40;

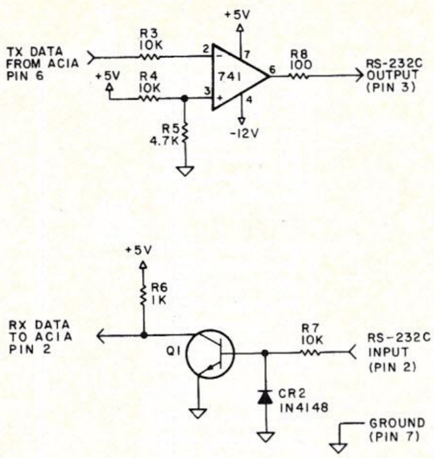



*Fig. 1. RS-232C/TTL level conversion.*

these pins are the data I/O ports that connect to the outside world. (I will give much more information regarding the PIA in a future installment.)

**Installing an ACIA**

Installing an ACIA takes a little more work, but is still useful. To take advantage of the nine pins prewired for either a PIA or an ACIA, the ACIA socket must be installed so pin 1 of the ACIA goes in the hole that would otherwise be used for pin 6 of the PIA. When installed in this way, the data bus and clock connections fit perfectly into the connections already established for the PIA.

The remaining ACIA pins must be connected by hand as follows:

Pin 1 to ground.

Pins 2 and 6 are the serial data input and output, respectively. Both of these are TTL-compatible, whereas most serial I/O devices will use either current loop or EIA RS-232C connections. This means that we must build an interface circuit that will convert between the TTL voltage levels used by the ACIA and either the voltage levels of an RS-232C connection or the current levels of a current loop connection. This interface circuitry, which I discussed in Kilobaud Klassroom No. 13 (October 1978, p. 46), can be built in the wire-wrap area of the board.

Fig. 1 shows the level conversion circuits needed to interface to an RS-232C terminal. The top circuit converts the TTL output of the ACIA to RS-232C levels of between −3 and −15 volts for a 1, and between +3 and +15 volts for a 0. It uses an inexpensive 741 op amp to provide the required inversion and to provide positive and negative output voltages. (Although a negative voltage supply is obviously needed to provide a negative output, the current is negligible and so a 9-volt transistor radio battery will provide enough current for even extensive testing and experimenting.)

The bottom circuit is used in the opposite direction. Negative voltages from the RS-232C port turn off the transistor and provide a high level to the ACIA for a 1, while positive input voltages turn on the transistor and provide an output near 0 volts for a 0. Any inexpensive npn silicon transistor can be used in this circuit.

Pins 3 and 4 require a clock pulse at a frequency that is a multiple of the baud rate. The clock frequency can be either the same as the baud rate, 16 times the baud rate or 64 times the baud rate, depending on how the ACIA is programmed. But most designers, wishing to stay compatible with UARTs that require a frequency of 16 times the baud rate, use the same multiple with the ACIA. For example, for transmission at 300 baud, a frequency of 4800 pulses per second would be required.

Fig. 2 shows two common ways of generating this clock frequency. Fig. 2a uses a 555 timer IC as an oscillator to provide a frequency equal to 16 times the desired baud rate. Since the circuit frequency is set by an RC network, careful adjustment of potentiometer R2 is required, and even then the frequency may drift with time. But this circuit is still popular because it is inexpensive (though a frequency counter is needed for adjustment).

Several baud rate generator ICs that generate the same frequencies by using a crystal oscillator as a reference are available. One popular device is the Motorola MC14411. As shown in Fig. 2b, this circuit uses a 1.8432 MHz crystal, which connects directly to an on-chip oscillator to provide the crystal-controlled reference. Inside the 14411 is a series of counters that divides the crystal frequency to provide a number of simultaneous baud rate output signals. For example, the 300 baud rate output on pin 7 comes from a counter that divides the 1.8432 MHz crystal frequency by 384 to produce exactly 4800 pulses per second; this is equal to 16 times 300.

This circuit will provide almost any baud rate we might need (including some not shown in Fig. 2b), but it has the disadvantage of requiring a $10 IC, as well as a $5 crystal. If we don't need 110 baud, then the circuit of Fig. 3 saves the price of the crystal by using the computer's own clock as the crystal reference. (But this circuit will only work if the 6802 is using a 3.579 MHz color TV crystal for the clock.)

When a 3.579 MHz crystal is used as the 6802 clock, this frequency is divided by the 6802 to provide an Enable signal of 3.579/4 MHz, or .89475 MHz. This is just a little less than half of 1.8432 MHz (about three percent less, to be exact). If we send this signal to pin 21 of the 14411 instead of the 1.8432 MHz signal that would normally be there from the crystal, the 14411 will generate outputs just slightly less than half the normal ones. For example, pin 7 will provide an output of just under 150 baud, instead of the normal 300 baud. So if we need 300 baud, we simply move over to pin 5, which now provides just under 300 baud instead of its customary 600 baud. Although these baud rate signals are about three percent low, this is still within the normal tolerance of serial terminals.

ACIA pins 8, 10 and 12 should be jumpered to +5 volts. The most convenient location is the hole that would otherwise go to pin 22 of a PIA.

Pin 9, $\overline{CS2}$, should be connected to pin 11 of the 74LS138 address decoder at the small round pad next to it.

ACIA pin 11, RS, connects to address bit A0 at the hole that would normally connect to pin 36 of a PIA in that IC position.

Pin 13, $R/\overline{W}$, should be connected to the hole that would otherwise go to pin 21 of a PIA.

Finally, pins 23 and 24 should be grounded to pin 1 of the PIA socket. (If, however, a serial port with handshaking is needed,

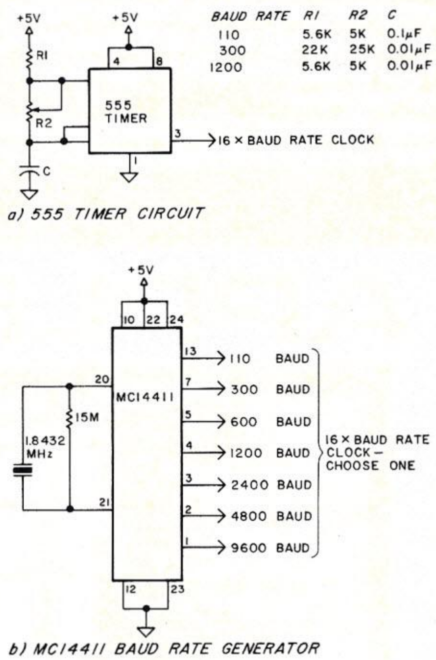



*Fig. 2. Two circuits for generating a 16x baud rate clock for an ACIA.*

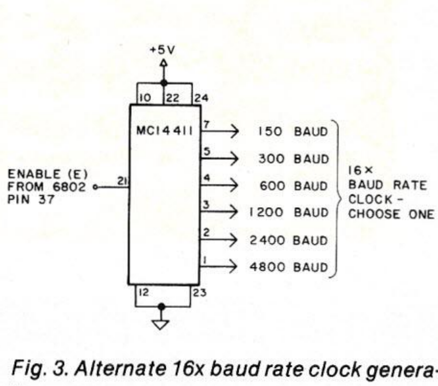



*Fig. 3. Alternate 16x baud rate clock generator.*

```
-12V          741-4
5V            *PIA2-22    ACIA-10    ACIA-12    ACIA-8    MC14411-10
              MC14411-22    MC14411-24    R4-1    R6-1
741+          741-3    R4-2    R5-1
741-          741-2    R3-2
741OUT        741-6    R8-1
8V            741-7
A0            *PIA2-36    ACIA-11
ACIASEL'      *74LS138-11    ACIA-9
BAUDCLOCK     ACIA-3    ACIA-4    MC14411-3
EIA-IN        EIA-PIN-2    R7-2
EIA-OUT       EIA-PIN-3    R8-2
ENABLE        *PIA2-25    MC14411-21
GROUND        *PIA2-1    ACIA-1    ACIA-23    ACIA-24    CR2-2    EIA-PIN-7
              MC14411-12    MC14411-23    Q1-E    R5-2
Q1BASE        CR2-1    Q1-B    R7-1
R/W'          *PIA2-21    ACIA-13
RXDATA        ACIA-2    Q1-C    R6-2
TXDATA        ACIA-6    R3-1
```

*Table 1. Wiring list for adding an ACIA, baud rate generator and serial RS-232C port.*

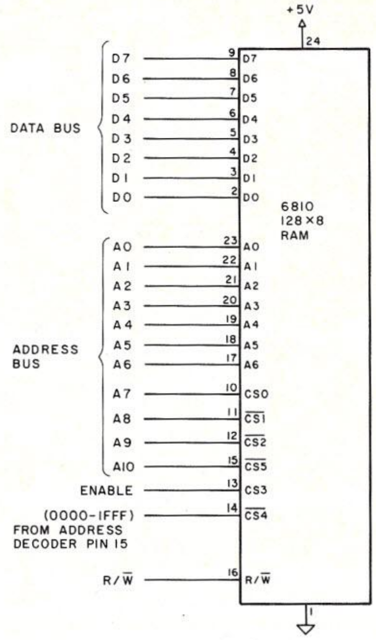



*Fig. 4. 128-byte RAM addition.*

then pin 24, $\overline{CTS}$, is used for that purpose. When pin 24 is low, the ACIA will output; when it is high, the ACIA will stop outputting and will wait.)

Table 1 is a wiring list that shows all the connections needed to install a complete serial port using the circuitry of Figs. 1 and 3. The connection points marked with a star are existing points in the computer to which you can connect to get the desired signal. For example, address bit A0 is listed as

A0 *PIA2-36 ACIA-11

which means that A0 is present on pin 36 of the PIA2 socket and should be connected to pin 11 of the ACIA.

Photo 1 shows how this addition mounts on the printed circuit board. The 6850 ACIA is in the top right corner, with the 14411 right under it. The eight-pin IC under that is the 741 op amp, and the rest of the RS-232C interface is just to its left. In this case, the 25-pin RS-232C connector is on the other end of a six-foot cable, but it could have been attached to the edge of the board with a hot-melt glue gun.

### Adding 128 Bytes More RAM

The basic 128 bytes of RAM inside the 6802 is enough for many applications, but sometimes additional RAM is useful. There are two easy ways to add a little more RAM. The easiest—requiring just one IC—adds another 128 bytes of RAM using an MCM6810 RAM for about $5 (Fig. 4). The wiring list for adding this modification is

shown as Table 2.

No additional circuitry is needed since this IC has six chip select (CS) inputs. To enable the 6810, CS0 and CS3 must be high, while $\overline{CS1}$, $\overline{CS2}$, $\overline{CS4}$ and $\overline{CS5}$ must be low. In this circuit, these chip selects work like this:

$\overline{CS4}$ connects to pin 15 of the 74LS138 address decoder (shown in Fig. 9 of the June installment). This pin goes low only on valid memory addresses beginning with 000 in bits A15, A14 and A13; this translates to the entire range from 0000 to 1FFF (in hexadecimal).

CS0, $\overline{CS1}$, $\overline{CS2}$ and $\overline{CS5}$ connect to address bits A7 through A10; together with $\overline{CS4}$, then, any memory address such as

000x x000 1xxx xxxx

(where x stands for a don't-care bit, which could be either 0 or 1) will select this RAM IC. If A12 and A11, the two don't-care bits on the left, are 00, the 6810 takes on addresses from 0080 through 00FF, which is the 128 bytes just above the 128 bytes already in the 6802, giving us a total of 256 bytes from 0000 through 00FF. This is a useful address range, since this first group of 256 locations (called page 0) is especially easy and quick to address in a 6800 or 6802 processor.

(Because of incomplete address decoding, if those two don't-care bits are nonzero, this RAM also responds to addresses 0880, 1080 and 1880, but this doesn't conflict with any other address assignments.)

Finally, CS3 connects to the Enable signal coming from the 6802. This is a timing signal that makes sure that the 6810 is selected only when valid data actually exists on the data bus. As shown in the waveforms in the June installment, data is present on the data bus at the end of the E signal and should normally be grabbed off the bus at that time.

When data is coming from the outside back to the 6802, the processor does this automatically; but whenever data is being sent from the 6802 out to some other device such as a memory IC or I/O device, then this device must be told when to get the data. This is why it wasn't necessary to send the E signal to the 2716 EPROM (which never gets any data from the data bus), but has to be sent to the PIA, ACIA and all RAM.

| | | |
|---|---|---|
| 5V | 6810-24 | |
| A0 | *6802-9 | 6810-23 |
| A1 | *6802-10 | 6810-22 |
| A2 | *6802-11 | 6810-21 |
| A3 | *6802-12 | 6810-20 |
| A4 | *6802-13 | 6810-19 |
| A5 | *6802-14 | 6810-18 |
| A6 | *6802-15 | 6810-17 |
| A7 | *6802-16 | 6810-10 |
| A8 | *6802-17 | 6810-11 |
| A9 | *6802-18 | 6810-12 |
| A10 | *6802-19 | 6810-15 |
| ADDR0000' | *74LS138-15 | 6810-14 |
| D0 | *6802-33 | 6810-2 |
| D1 | *6802-32 | 6810-3 |
| D2 | *6802-31 | 6810-4 |
| D3 | *6802-30 | 6810-5 |
| D4 | *6802-29 | 6810-6 |
| D5 | *6802-28 | 6810-7 |
| D6 | *6802-27 | 6810-8 |
| D7 | *6802-26 | 6810-9 |
| ENABLE | *6802-37 | 6810-13 |
| GROUND | 6810-1 | |
| R/W' | *6802-34 | 6810-16 |

*Table 2. Wiring list for adding an MCM 6810 RAM.*

**Adding 1K More RAM**

A slightly more ambitious project is to add an entire 1K of RAM with just three chips, using the circuit of Fig. 5 for a total cost of about \$15. These three ICs are shown in Photo 1, just to the left of the 14411 baud rate generator.

The 74LS138, labeled SEL2 in Fig. 5, is an additional address decoder (not the same as the 74LS138 already on the board). Its primary function is to gate the Enable clock with the address 2000–3FFF signal from the main 74LS138 decoder, so that the two 2114 RAMs get a low $\overline{CS}$ signal only when the E signal is high and the 2000–3FFF select signal is low. This is absolutely necessary to make sure that the RAM grabs data off the data bus at the right time when storing. Although this could just as easily

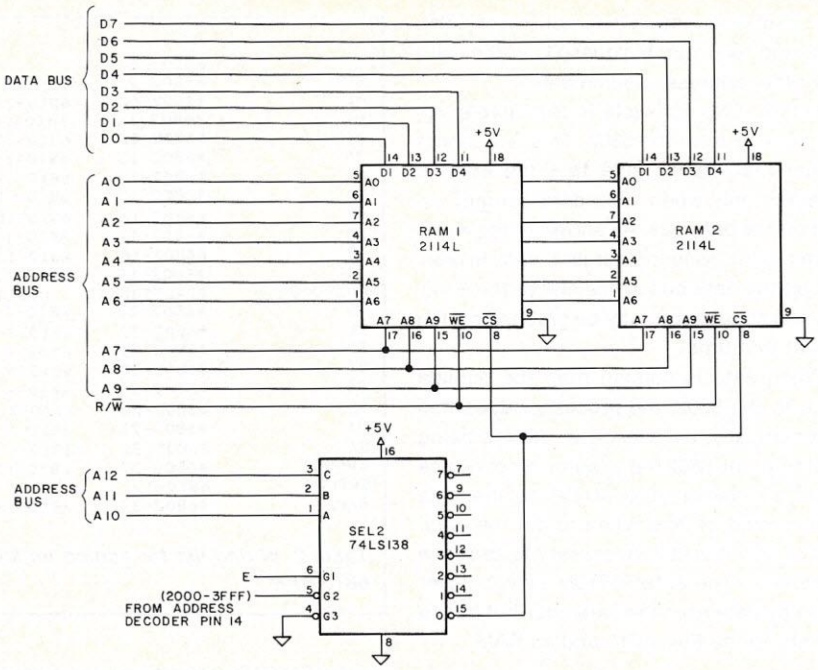



*Fig. 5. 1K RAM addition to the basic computer.*

be done with a 74LS00 gate at slightly less cost, the 74LS138 does a better job since it also decodes the addresses better—it requires that address bits A12 through A10 also be 0, so that the 2114 RAM only responds to the 1K address range from 2000 to 3FFF.

Though we can really make do without this, it simplifies things if we ever decide to add another pair of 2114s, since the eight outputs of SEL2 divide up the 8K address range from 2000 through 3FFF into eight 1K ranges.

We could add another 1K of memory by wiring up still another pair of 2114s in the same way, except that the $\overline{CS}$ pin on this pair would go to pin 14 of SEL2. If it weren't for loading the 6802 address and data buses (and lack of room on the board), we could thus add a total of 8K memory. In practice, though, we would find that having much more than 1K of memory would overload the buses and start causing other problems unless we buffered them. Since this would turn the computer into something other than what we started with, let's not consider that further.

Table 3 is a wiring list that shows all the wiring needed to add this 1K memory expansion to the computer. As before, connection points marked with a star are points in the computer where the required signal can be obtained.

### A Few Extra Bits

If you really don't need a full PIA or ACIA (or have already added one and find you still need an extra bit or two of input or output), then all you need is a flip-flop or three-state buffer. The idea is to use readily available, inexpensive ICs (Radio Shack has them), but take advantage of all the extra unused 74LS138 outputs.

Fig. 6 shows how the 74LS367 hex three-state buffer can be added to provide six input bits. The six inputs come in from the left, while the six outputs on the right go to six bits of the data bus.

The 74LS367 has six three-state buffers divided into two groups—the top two are turned on when the signal called Select 2 goes low, while the bottom four go on when Select 1 goes low.

If we tie these select signals to the unused outputs of the 74LS138 address decoders, we can select either group of buffers whenever the specified address is used in a program. (Normally, the two select signals would be connected together to the same 74LS138 output, but they could go to two different pins.)

For instance, suppose that both select pins are connected to pin 12 of the main 74LS138 address decoder (shown in Fig. 9 of the June installment). This pin goes low whenever any address in the range of 6000–7FFF is encountered in a program.

Whenever the computer program does any read from any of these addresses, this signal turns on the 74LS367 input buffers, and the 6802 does a read from these six input lines. Hence, this makes a simple six-bit input port.

Since the R/$\overline{W}$ signal isn't used anywhere in this circuit, this input port will be selected regardless of whether we read from, or write out to, one of the addresses in the 6000–7FFF range. If we are writing to such an address, then the 6802 will be putting data on the data bus at the same time as the 74LS367 does so, and this will result in garbage on the bus (as well as undesirable loading on the bus as two drives try to force it in different directions). Thus, if we use a simple port such as this one, we must make sure that our program only reads from it, never writes to it.

Fig. 7a shows the correct way to wire a one-bit output port. One of the data bits from the data bus goes to the data input of a type D flip-flop such as a 74LS74, and a select signal from an address decoder (the 74LS138) goes to the clock. At the end of the select pulse—the rising edge shown in the diagram—the data from the data bit is clocked into the flip-flop.

Though this is the theoretically correct way to wire an output bit, it has two problems. First, the D input of the flip-flop adds an extra load to the data bus, which may

| | | | | |
|---|---|---|---|---|
| 5V | RAM1-18 | RAM2-18 | SEL2-16 | |
| A0 | *6802-9 | RAM1-5 | RAM2-5 | |
| A1 | *6802-10 | RAM1-6 | RAM2-6 | |
| A10 | *6802-19 | SEL2-1 | | |
| A11 | *6802-20 | SEL2-2 | | |
| A12 | *6802-22 | SEL2-3 | | |
| A2 | *6802-11 | RAM1-7 | RAM2-7 | |
| A3 | *6802-12 | RAM1-4 | RAM2-4 | |
| A4 | *6802-13 | RAM1-3 | RAM2-3 | |
| A5 | *6802-14 | RAM1-2 | RAM2-2 | |
| A6 | *6802-15 | RAM1-1 | RAM2-1 | |
| A7 | *6802-16 | RAM1-17 | RAM2-17 | |
| A8 | *6802-17 | RAM1-16 | RAM2-16 | |
| A9 | *6802-18 | RAM1-15 | RAM2-15 | |
| ADDR2000' | *74LS138-14 | SEL2-5 | | |
| D0 | *6802-33 | RAM1-14 | | |
| D1 | *6802-32 | RAM1-13 | | |
| D2 | *6802-31 | RAM1-12 | | |
| D3 | *6802-30 | RAM1-11 | | |
| D4 | *6802-29 | RAM2-14 | | |
| D5 | *6802-28 | RAM2-13 | | |
| D6 | *6802-27 | RAM2-12 | | |
| D7 | *6802-26 | RAM2-11 | | |
| ENABLE | *6802-37 | SEL2-6 | | |
| GROUND | RAM1-9 | RAM2-9 | SEL2-4 | SEL2-8 |
| R/W' | *6802-34 | RAM1-10 | RAM2-10 | |
| RAMSEL' | RAM1-8 | RAM2-8 | SEL2-15 | |

*Table 3. Wiring list for adding 1K of RAM with two 2114s.*

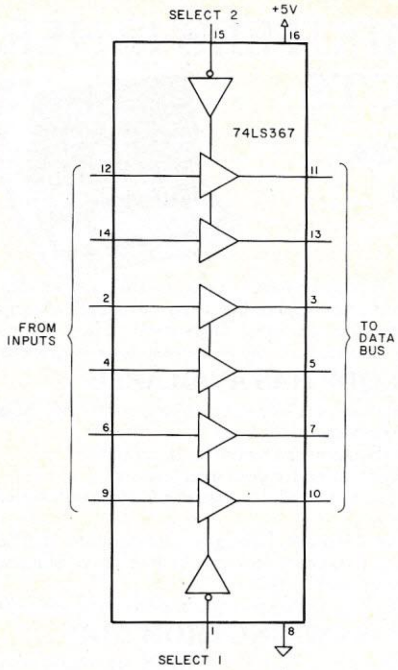



*Fig. 6. Simple input port using a 74LS367.*

cause problems if the data bus is already heavily loaded.

Second, since the data bit should be entered into the flip-flop at the end of the Enable signal, this signal should in some way be part of the select signal going to the clock input. Thus, the outputs of the SEL2 decoder 74LS138 IC shown in Fig. 5 would be suitable, but the outputs of the main 74LS138 would not.

If we don't use the data bus, then the timing is not as critical. Fig. 7b shows an alternative idea. Here, the select signal from the 74LS138 goes to the clock input of a JK flip-flop such as the 74LS73, while the J and K inputs both go straight to +5 volts. With this connection, the flip-flop will toggle every time it gets a clock pulse. Hence, we can flip the flip-flop simply by selecting the correct address (so the 74LS138 outputs a pulse).

The trouble with this circuit is that we can never be quite sure whether the flip-flop is on or off at any particular time, since the computer has no way of knowing whether it started out set or reset when the power first came on. In some cases this doesn't matter—in generating music, for instance.

But if it does, then we could disconnect the CLR pin from +5 volts (which was disabling it) and connect it instead to the reset pin of the 6802. Since this pin is always low when the system is first turned on, this will guarantee that the flip-flop will always be reset when we first power up the system.

Fig. 7c is another possibility. Here we disable the D input and clock (or the JK and clock inputs of a JK flip-flop) by connecting them to +5 volts. Instead, we take two different select outputs from the 74LS138 and connect one to the PS (preset) input, and the other to the CLR (clear) input of the flip-flop. Now we can set or reset the flip-flop explicitly.

For example, suppose the PS input goes to the 6000–7FFF output of the address decoder (pin 12), while the CLR input goes to the 4000–5FFF output (pin 13). Any time we access any address in the 6000 range, the flip-flop gets a low PS pulse and sets; if you access any address in the 4000 range, it will reset.

Since we only have a few unused outputs on the 74LS138, we obviously cannot connect too many such flip-flops, but with four decoder outputs we could trigger three flip-flops simply by connecting all three resets to the same output and connecting the three sets to three outputs. Now we don't have completely independent control over the flip-flops, since they will all reset together. But this may still be quite useful.

### From Hardware to Software

Having seen how to build, test and even expand our control computer, it's time to examine how to program it. Although the computer can run programs written in Tiny BASIC when properly expanded, let's keep in mind its primary purpose—to be a dedicated control computer. That requires that we program it in machine or assembly language.

So let's discuss machine- and assembly-language programming. Rather than start at the very beginning, we will assume that all our readers have some familiarity with programming in BASIC.

Since 6802 machine language is identical with that of the 6800, this discussion applies to both processors.

### 6802 Internal Structure

When programming in BASIC, we tend to think in terms of the job to be done, rather than the way in which it is being done. Machine- and assembly-language programming is different—you must constantly think of the hardware that is doing the program. Thus, you can write a BASIC program without even knowing what computer it will be used on; but to write programs in machine or assembly language, you must know what is inside the computer and—to some extent—how it works. This makes programming tougher, but also more fun and more challenging.

Fig. 8 shows a simplified view of what is inside the 6802 (or 6800) processor. (A few extra registers, as well as the 128-byte RAM, are not shown, simply because they do not concern us at this point.)

Within the 6802, the hardware consists primarily of a set of various registers, plus an eight-bit transfer bus that interconnects them and allows data or addresses to go back and forth within the processor. A register is essentially a group of storage elements—flip-flops or dynamic memory cells—that holds a binary number. While a program is running, numbers are constantly being moved back and forth between these registers.

Some of these transfers are managed by the 6802's internal control circuitry and are completely out of our control; other transfers are directly controlled by the program instructions we write.

Some of the registers hold eight bits, some 16 bits. One (the condition code register at the top) only holds six. In any case, all of these are interconnected by an eight-bit transfer bus, so that each half of a 16-bit register is connected to the bus separately. When a 16-bit number is transferred from one place to another, it is moved in two eight-bit pieces. This will help to explain why some instructions are faster than others.

Two of these registers are called accumulators; they are the A accumulator and the B accumulator. An accumulator is a type of register that can not only hold a number, but can also do some additional operations on it, such as addition or subtraction. Both 6802 accumulators hold eight-bit numbers. Virtually all arithmetic operations in a 6802 system are done in one of the accumulators.

Accumulators are generally the workhorses of a computer. There are instructions for loading a number from memory into an accumulator, for adding or subtracting numbers in an accumulator, or for storing a number from an accumulator back into memory. In addition, numbers in an accu-

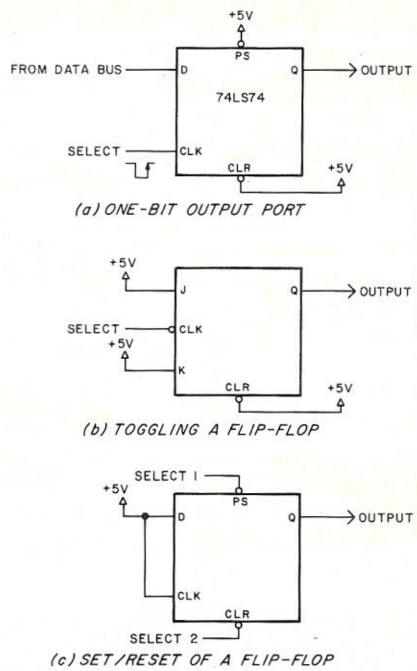



*Fig. 7. Three simple output ports.*

mulator can be tested in various ways, moved from one accumulator to another or shifted left or right.

The six-bit register is the condition-code register. Though it is called a register for convenience, it really is composed of several different parts, each of which holds just one bit. These bits are used to indicate whether the result of a previous operation had some specific trait, such as being zero, nonzero or negative. One of these bits is also used to control the interrupt system. (Although this register only has six bits, when we transfer its contents into an eight-bit register or into memory, we pick up an extra two bits along the way. These sometimes fool us into thinking this register really has eight bits when it does not.)

Another useful register is the index register. Though the index register can be used for a variety of purposes, its most common use is to hold and manipulate addresses; thus it is a 16-bit register so that it can hold a full 16-bit address.

The index register can be loaded from memory; its contents can be incremented or decremented (increased or decreased by 1) or stored back into memory. In addition, the contents of the index register can be used as a pointer to point to a specific location in memory which will be accessed at some point.

Another 16-bit register is called the stack pointer. This register points to the next empty location in the stack; the stack, in turn, is a memory area set aside for temporary storage of data and addresses. A common use for the stack is to hold the return address from a subroutine—in terms of BASIC, this means that when a go-to-subroutine instruction (such as GOSUB) is executed, the computer places into the stack the address of the instruction it should return to when the subroutine is finished. The stack pointer register is simply a pointer used to indicate which part of the stack has been filled up and which part is still empty. As numbers and addresses are put into or taken from the stack, the pointer keeps changing, so that it always points to the next empty location in the stack.

The last 16-bit register of concern to us is the program counter. Actually, the programmer doesn't have much control over the program counter, since there are no instructions that specifically let you manipulate that register. But the program counter is important since it keeps tab on the next instruction to be performed in our program. It is affected by transfer instructions (which are similar to BASIC's GOTO, GOSUB or RETURN).

Though a few more registers are inside the 6800 or 6802, these are used internally and are not normally used by the programmer.

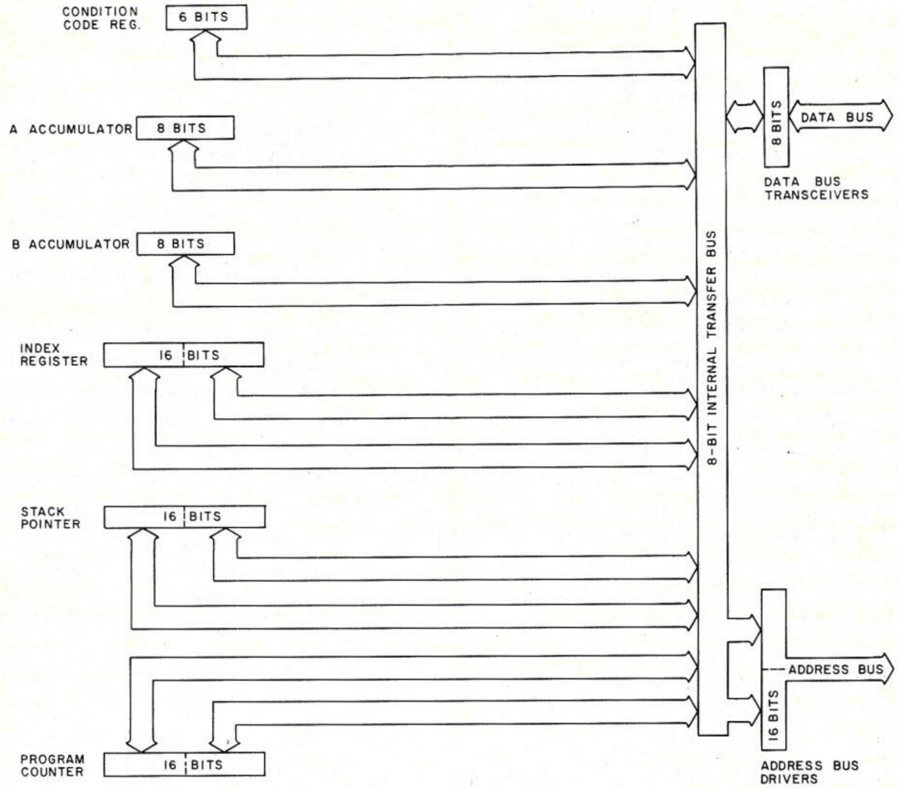



*Fig. 8. The inside of the 6802.*

If you are familiar with other processors, you may note that the 6802 registers are different from those you are familiar with. The register structure of a processor—and the instructions it has to use them—distinguishes one microprocessor from another. Some processors, such as the 8080 or Z-80, have many different registers; others, such as the TMS-9000 or the SC/MP, have few.

Actually, the number of registers does not really determine whether a processor is powerful or not, since those processors that have few registers tend to have instructions that allow them to do operations that are optimized to work without them.

Though various articles have compared different microprocessor structures (see, for example, the excellent articles by Hal T. Gordon entitled "Instruction Sets Examined and Compared" in the March and April 1980 issues of *Microcomputing*), different programmers will prefer different processors. Many programmers like the 6800 and 6802 because it is simple yet powerful; the fact that its architecture is more traditional rather than unusual (such as the TI TMS or the Signetics 2650) may also have a lot to do with that.

### Computer Languages

Computer programs must be written in a special way so that the computer will be able to understand them. This generally requires that we use a certain vocabulary and a set of grammar rules, much like a human language. Thus we get the term "computer language" to describe the way a program is written.

There are essentially three types of computer languages:

•Machine language is the only language that the computer can really understand. Since the computer works with binary digits—ones and zeros—the machine-language program is written in these as well. When a machine-language program is entered into the computer, it can be immediately performed.

Note that machine language is directly tied into the hardware of the processor. Thus, different processors will have different machine languages, and a program written for one will not run on another (unless you intentionally copy the language of the other).

•Assembly language is one step above machine language. In many ways, it is similar to machine language in that generally one instruction in the assembly-language program is equivalent to one instruction in the machine-language program. The difference is that that instruction consists of numbers in machine language, whereas it may consist of simple letter codes in assembly language.

Since assembly language cannot be directly understood by the computer, it must first be translated into machine language. But since the two languages are so similar, that translation is fairly simple. For short programs you can do it yourself; for longer programs this translation is done by still an-

other computer program called an assembler.

Since assembly language is so close to machine language that it is translated on a more-or-less one-to-one basis (one assembly-language instruction becomes one machine-language instruction), different machine languages must result in different assembly languages. An assembly-language program written for one processor will also not run on another.

•Higher-level languages, also sometimes called problem-oriented languages, are at the top of the list. These languages are completely different from machine or assembly language and generally require quite extensive translation into machine language before they can be executed on the computer. This translation could be done manually, but even for short programs this is not practical; hence, translation is usually done by a translator program called a compiler or an interpreter.

Higher-level languages are aimed at a particular application, rather than a particular computer. Since extensive translation is required anyway, there is no need to tailor the language to fit a machine. A language such as BASIC is similar whether it runs on an IBM 370 or a TRS-80. Obviously, though, the translator for the two computers will be different, and the resulting machine language will also be much different.

The higher-level languages are oriented toward specific kinds of problems. Languages such as BASIC, FORTRAN, ALGOL or MAD are at their best with numeric problems from math or engineering; COBOL or RPG might best be suited for business applications; while languages such as LISP, SNOBOL or LOLITA might be good for strings.

(There have been hundreds of higher-level languages in the few short decades of computers, including some with interesting names such as JOVIAL, MADCAP, ADAM and BASEBALL. If you are interested in a readable history of the subject, I strongly recommend *Programming Languages: History and Fundamentals* by Jean E. Sammet, published by Prentice-Hall, Inc., in 1969 and readily available in larger libraries.)

There are many exceptions. Some smart assemblers allow a single assembly-language instruction to translate into many machine-language steps. And in some computers, the translator for a higher-level language is built-in so that it almost looks as though this higher-level language is the only language the machine can understand. (That is certainly how a TRS-80 must appear to the beginner!)

In a higher-level language, we are concerned with the job to be done, and not with the mechanics of how it is to be achieved. For example, to add two numbers in BASIC, we simply say

```
C=A+B
```

In machine or assembly language, we must be concerned with small details, such as where in memory A, B and C are to be found and what part of the processor will be used to perform the addition.

Remembering that memory locations are referred to by their addresses, we might have A in location 100, B in location 101 and C in location 102. Then the C = A + B program would have to be broken down into smaller steps such as:

1. Take the number in location 100 and bring it into accumulator A.
2. Add the number in memory location 101 to whatever is already in accumulator A.
3. Store the number from accumulator A into memory location 102.

If the three numbers are larger than what can be stored in just a single eight-bit memory location, then they might be split up among several consecutive locations, and the machine- or assembly-language program to perform a simple C = A + B equation might consist of dozens or even hundreds of steps. ■

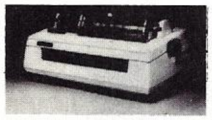
# WE WILL NOT BE UNDERSOLD

## 16K MEMORY UPGRADE KITS 2 for $85 $45

for TRS-80*, Apple II, (specify): **Jumpers** **$2.50**

## PRINTERS

### NEC Spinwriter



**Letter Quality High Speed Printer**
Includes TRS-80* interface software, quick change print fonts, 55 cps, bidirectional, high resolution plotting, graphing, proportional spacing: R.O. **$2579**

R.O. with Tractor Feed **$2679** KSR with Tractor Feed **$2995**

**779 CENTRONICS TRACTOR FEED PRINTER** **$969**
Same as Radio Shack line printer I

**737 CENTRONICS FRICTION & PIN FEED PRINTER** **$799**
9 × 7 matrix

**730 CENTRONICS FRICTION & PIN FEED PRINTER** **$629**
7 × 7 matrix Same as Radio Shack line printer II

**P1 CENTRONICS PRINTER** **$269**
Same as Radio Shack quick printer

**PAPER TIGER** (IP440) **$939**
Includes 2K buffer and graphics option

**TI-810** Faster than Radio Shack line printer III
Parallel and serial w/TRS-80* interface software
with upper and lower case and paper tray **$1599**

**OKIDATA Microline 80** Friction and pin feed **$559**
Tractor Feed, friction, and pin feed **$679**

**EATON LRC 7000 +** 64 columns, plain paper **$299**

**ANADEX** **DP-9500** **$1359** **DP-8000** **$825**

## DISK OPERATING SYSTEMS

**PATCHPAK #4 by Percom Data** **$ 8.95**
**CP/M**® for Model I, Zenith **$145** • for Model II, Altos **$169.00**
**NEWDOS Plus** **40 track** **$ 99.00**
**NEWDOS 80** **$135.00**

## ACCESSORIES

**HEAD CLEANING DISKETTE:** Cleans drive Read/Write head in 30 seconds. Diskettes absorb loose oxide particles, fingerprints, and other foreign particles that might hinder the performance of the drive head. Lasts at least 3 months with daily use. Specify 5¼″ or 8″. **$20 ea/$45 for 3**

**FLOPPY SAVER:** Protection for center holes of 5¼″ floppy disks. Only 1 needed per diskette. Kit contains centering post, pressure tool, tough 7-mil mylar reinforcing rings. Installation tools and rings for 25 diskettes. **$11.95**
**Re-orders of rings only** **$ 7.95**

**EXTERNAL DATA SEPARATOR:** Eliminates data separation problems (crc). Improves reliability. This plug in unit comes fully assembled and tested. **$29.95**

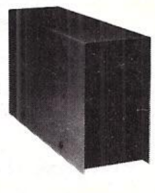
**RS232** **$84.00**

**DISK-DRIVE EXTENDER CABLES:** Fits all mini-disk drives. **$16.95**

**SIX (6) PRONG ISOLATOR** **$54.00**
**AC FILTER/6 PRONG POWER STRIP** **$39.00**
**DISK DRIVE CABLES:** 2 drive **$29.00** 4 drive **$35.00**
**DUST COVERS:** TRS-80/Apple **$ 7.95**
**PLASTIC DISKETTE HOLDER** **$ 8.00**
**RF MODULATOR:** Adapts video to TV **$35.00**
**TRS-80 & OTHER MYSTERIES** **$18.95**

**CAT MODEM** Originate and answer same as Radio Shack Telephone Interface II **$148**
**LEEDEX MONITOR Video 100** **$119**
**ZENITH Color Monitor** **$379**
**SANYO Model VM 4509 Monitor** **$179**

## DISK DRIVES $314



40 track, 102K Bytes. Fully assembled and tested. Ready to plug-in and run the moment you receive it. Can be intermixed with each other and Radio Shack drive on same cable. TRS-80* compatible silver enclosure. 90 day warranty. One year on power supply. **External card edge included.**

| | | |
|---|---|---|
| **FOR TRS-80*** | | |
| CCI-100 | 5¼″, 40 Track (102K Bytes) for Model I | **$314** |
| CCI-280 | 5¼″, 80 Track (204K Bytes) for Model I | **$549** |
| CCI-800 | 8″ Drive for Model II (½ Meg Bytes) | **$795** |
| **For Zenith Z89** | | |
| CCI-189 | 5¼″, 40 Track (102K Bytes) add-on drive | **$394** |
| Z-87 | Dual 5¼″ add-on drive system | **$995** |

**DISKETTES —** Box of 10 (5¼″) - with plastic library case **$24**
8″ double density for Model II (box of 10) **$36**

## COMPLETE SYSTEMS

**ALTOS 64K, DD, SS, 2-Drive, 1MB** **$3995**
**TRS-80* Model II-64K** **$3499**
**TRS-80* LEVEL II-16K with keypad** **$689**
**TRS-80* Expansion Interface** **$249**
**APPLE 16K** **$989**
**HEWLETT PACKARD HP-85** **$2999**
**ZENITH Z89, 48K** all-in-one computer **$2555**
**ZENITH Z19** **$740**
**TELEVIDEO** **912B $745** **920B $769**
**ATARI 400 $489** **ATARI 800 $769**
**APF** **M1000 $99** **IM-1 $499**
**MATTEL INTELLIVISION** **$249**

## SOFTWARE FOR THE TRS-80* Software w/Manual

**INTELLIGENT TERMINAL SYSTEM ST-80-III BY LANCE MIKLUS:** Enables a TRS-80* to act as a dial-up terminal on any standard time sharing network. Provides a TRS-80* with control key, ESC Key, Repeat Key, Rub Out Key, Break Key, full upper and lower case support, selectable printer output and program selectable transmission rates **$139**

**CCA-DATA MANAGEMENT SYSTEM:** Automate your information processing tasks. You can create a file of customer information; quickly and easily add, delete or update records; search a file; keep a file in order of the value in any field; and print records and labels in any desired sequence or from just a part of a file. Requires 32K TRS-80 and one drive. **$74.95**

**S & M SYSTEMS**
**INSEQ-80™:** Indexed Sequential Access Method (ISAM) for the TRS-80 Model I. A must for anyone writing business programs. Eliminate wasted disk space from direct record processing. Split second access to any record. Access data records instantly via alpha/numeric "key" eg. Part NR, zip code or sequentially in ascending key sequence. Add/modify records in any order. Access up to three files per program — Files may be spread over multiple disks. Machine language processing from your basic program. Utility program to convert direct files to INSEQ-80 format. **$49.95**

**FULLY INTERACTIVE ACCOUNTING PACKAGE:** ISAM (INSEQ-80) based. Includes General Ledger, Accounts Payable, Accounts Receivable and Payroll. System runs "stand alone" or "co-ordinated G/L" at users option. Based on Osborne accounting method. Requires 32K, TRS-80, 2 or 3 drives. N/A CA.
**General Ledger** **$99**
**Accounts Receivable** **$99**
**Accounts Payable** **$99**
**Payroll** **$99**
Osborne books: Req'd as additional documentation **$20 ea**

**INVENTORY** Requires 32K, TRS-80, 1 drive **$125**

**INSORT-80:** Callable form BASIC via USR. Sorts "Random" Disk Files. "Disk" to "Disk" sort times — 350 records in 35 secs, 1000 records in 6 minutes, 3500 records in 12 minutes. Machine language processing. Up to 35 sort keys ascending/descending. Utility to build BASIC program. Runs under NEWDOS. **$49.95**

## CP/M BASED SOFTWARE for Zenith, Altos, Radio Shack, Apple Software w/Manual

**Z-80 SOFTCARD FOR APPLE:** Your key to future software expansion. Get the best of both worlds, Apple's 6502 and CP/M Z-80. Plug in the card and get a Z80. Supports Apple language card and all Apple peripherals. Comes with set of three manuals. **$339**

**CCI-TELNET VERSION 5:** A communication Package which enables microcomputer users to communicate both with Large Mainframes and other microcomputers. Extensive commands make it useful in many applications where communication between computers is necessary. Powerful terminal mode enabling user to save all data from a session on disk. Completely CP/M compatible. Multiple communication protocols supported. Able to transfer files in both directions without protocol where the other machine does not support any protocol. Extensive ON-SCREEN help. Source code provided. **$149**

**MICROPRO-WORD-STAR:** Menu driven visual word processing system for use with standard terminals. Text formatting performed on screen. Facilities for text paginate, page number, justify, center and underscore. User can print one document while simultaneously editing a second. Edit facilities include global search and replace. Read/Write to other text files, block move, etc. Requires CRT terminal with addressable cursor positioning. **$399**

**DEALER (NATIONAL/INTERNATIONAL) INQUIRIES INVITED** **Send for FREE Catalogue**

# The CPU SHOP

✓256

**TO ORDER CALL TOLL FREE 1-800-343-6522**
Massachusetts residents call **(617) 242-3361**
For detailed technical information, call 617/242-3361
Hours: 10AM-6PM (EST) M-F (Sat. till 5)
*TRS-80 is a Tandy Corporation Trademark ® Digital Research

5 Dexter Row, Dept. K10M
Charlestown, Massachusetts 02129

Massachusetts residents add 5% sales tax

Quantities on some items are limited

VISA®

master charge

## TRS-80 & OTHER NEEDS FILLED FOR LESS

+ + + COMPATIBLE DISK DRIVES WITH POWER SUPPLY AND CASE—120 DAY WARRANTY + + +

| | |
|---|---|
| • 40 TRACK (204,800 BYTE/DISK) USE BOTH SIDES, ANTI-CRIMP/POWER PROTECT | $329 |
| • 8 IN. DRIVE & P.S./CASE **$749** WITH P.S./CASE FOR 3 DRIVES | $929 |
| • 80 TRACK (204.8K BYTE) 90 DAY WARRANTY | $479 |

• 4-DRIVE CABLE **$28** •• 10 DISKS-5 IN. @ **$24**-8 IN. @ **$36** HARD CASE **$3** & **5**

••• BASE 2 PRINTER-(60 LINE/MIN, 72,80,96,120 or 132 CHAR. LINE BI-DIRECT DOT MATRIX IMPACT 96 CHAR ASCII, 15 BAUD RATES) **$499** 2K BUFFER, GRAPHICS, & TRACTOR OPTION AVAIL.•••

| | |
|---|---|
| • CENTRONICS 737 **$839** + + + + + CABLE @ **$25** | |
| • HARRIS SELECTRIC (WORD PROCESSING-TYPEWRITER & PRINTER) | $790 |
| • LOWER CASE FOR CENTRONICS 779/RADIO SHACK LINE PRINTER 1 -EASY INSTALL | $99.95 |
| • UPS (UNINTERRUPTIBLE POWER SUPPLY) PREVENT POWER DROP SURGE OR OUT? FROM | $195 |
| • CAT MODEM (ORIG/ANS) **$144** + + + + + + + + 16K MEMORY SET (200 NANO) | $44 |
| • 16K L2 RADIO SHACK COMPUTER SYSTEM | $649 |

• APPLE, ATARI, RADIO SHACK MODEL 1/2 HARDWARE/SOFTWARE DISCOUNTED A/R, A/P, G/L, P/R FOR **$360** or **$100** EA. (MODEL 1) & **$630** OR **$165** EA. (MODEL 2). APPLICATIONS INTERACT & ARE COMPLETE & PROFESSIONAL. WILL RUN ON OTHER COMPUTERS. THIS IS A SPECIAL INTRODUCTORY PRICE.

• ASK FOR FREE FLYER WITH OUR LOW PRICES—DEALER INQUIRIES INVITED MASS. RESIDENTS ADD 5% TAX—F.O.B. TEWKSBURY—FREIGHT EXTRA.

M/C, VISA OR CHECK ACCEPTED. TRS-80 IS A REG. TRADEMARK OF TANDY CORP.

✓140

**OMNITEK SYSTEMS** — 24 MARCIA JEAN DR., DEPT. K, TEWKSBURY, MA 01876 CALL 617-851-3156

# $250?

Our 50/80 INTERFACE lets you connect your Radio Shack TRS 80 - Model 1 to the IBM Electronic Typewriter Mod 50, 60 or 75. Aside from yielding the best looking printouts and listings you'll ever see, our system lets you center titles, underline words & phrases, justify numerical columns, indent text and more - from your own programs and most others. Right justified proportionally spaced typesetting is even possible with the IBM50! The most cost effective word processor ever. (this ad is an example) $250.00

SOFTWARE also available for any TRS80:

SUPERDIR - displays a menu like directory in DOS from which you can RUN or KILL any program, display updated FREE space & print the display, all with single key commands. In fast acting Z80 code, only $14.95 (DOS).

INMOD3 - easily used Z80 system program that can give any BASIC program professional keyboard entry. Blinking cursor, upper/lower case, user defined input length, repeat keys & single keystroke control codes. Makes INKEY$ obsolete; saves 1000 bytes over BASIC equivalent; for the rankest amateur! $14.95 (L2 or DOS).

INMOD3 Plus - same but works with Percom "Speak 2 Me": each character is spoken as entered! $19.95

* * * INTRODUCING our new MX80 FIRMWARE Interface and Modules. Software now in hardware form; utilizes the unused 2K lower mem. Write or call. VISA & MC. Dealers encouraged!

**MEDIAMIX**

PO Box 8775
Universal City, CA. 91608
213-475-9949
Micronet# 70250,321

## THE SPIKE-SPIKER™ with transient absorber and filters
### Computer Power Console

- ✓ Protects computer equipment from most power line transients
- ✓ Provides convenience of plugging all computer equipment into one unit and simply switching the equipment on and off in required sequential order
- ✓ Provide RF "hash" filtering between computer and motorized equipment in the computer system, home or office to help prevent interference
- ✓ Eliminates constant plugging and unplugging of power cords

The Spike-Spiker has 8 individually switched 120 VAC outlets divided into two rows of separate filtered circuits of 4 outlets each, main on/off switch, fuse, and indicator light. Prewired and ready to use!

Plug your CPU, interface, etc. in one filtered set of 4 outlets and your disks and printer in the other set of 4 outlets. This allows RF "hash" filtering to help prevent interference between the computer and its peripheral motorized equipment.

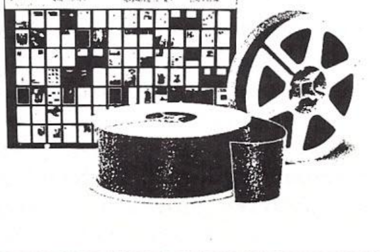
**$59.95**

**KALGLO®** ✓222 ELECTRONICS CO., INC.

COLONY DR. I.P.
P.O. BOX 2062
BETHLEHEM, PA 18001

OUT OF STATE
800-523-9685
215-865-0006

# this publication is available in microform



Please send me additional information.
**University Microfilms International**

300 North Zeeb Rd.
Dept. P.R.
Ann Arbor, MI 48106
U.S.A.

18 Bedford Row
Dept. P.R.
London, WC1R4EJ
England

## SAVE MORE THAN 20%

**NORTH STAR — INTERTUBE — MICROTEK**
**ZENITH — RCA-COSMAC — ITHACA**
**THINKER TOYS — GODBOUT — SUPERBRAIN**

*The smartest computers at the smartest price*

| FACTORY ASSEMBLED & TESTED | LIST | ONLY |
|---|---|---|
| HORIZON-1-32K-DOUBLE DEN | $2695 | $1994 |
| HORIZON-2-32K-DOUBLE DEN | 3095 | 2299 |
| HORIZON-2-32K-QUAD DENSITY | 3595 | 2699 |
| HORIZON-2-64K-QUAD + HARD DISK | 9329 | 7199 |
| HORIZON MEMORY ASSM | 16K 389 | 32K 579 |
| HORIZON MEMORY KIT | 16K 359 | 32K 535 |
| NORTH STAR HARD DISK 18 Mb | 4999 | 3939 |
| PASCAL FOR NORTH STAR ON DISK | 199 | 190 |
| Powerful NORTH STAR BASIC..The Best.................... | | FREE |
| 2 NORTH STAR SOFTWARE DISKS w/HORIZON............... | | FREE |
| NSSE 1-22 & P01 TERRIFIC PROGRAMS | ONLY | 10 |
| NORTHWORD 299 MAILMAN 239 INFOMAN | | 369 |
| RCA-COSMAC VP-111 99 RCA-COSMAC VP-711 | | 189 |
| COLOR! RAINBOW 385 CAT-100 1369 SPECTRUM | | 289 |
| ITHACA FRONT PANEL COMPUTER 64K | 3195 | 2695 |
| Z-8001 CPU CARD 16-bit ITHACA S-100 8Mb | | 1179 |
| ITHACA MEMORY 8/16-bit 64K | 995 | 845 |
| SEATTLE 8086 CPU 16 bit 10 x faster | | 556 |
| SEATTLE MEMORY 8/16 BIT 16K 4Mhz | | 356 |
| SSM KITS Z-80 CPU 221 VIDEO BRD VB3 4Mhz | | 412 |
| MEASUREMENT MEMORY 64K A & T 4mHz | | 599 |
| MEASUREMENT MEMORY 64K BANK SELECT | | 789 |
| ECONORAM XIV UNKIT 16K | 299 | 254 |
| CENTRAL DATA 64K RAM | 665 | 599 |
| DISCUS/2D A & T + CP/M | 1199 | 975 |
| THINKER TOYS HARD DISK 26 Mb | 4995 | 3995 |
| DISCUS/2+2 1.2 Mbytes A & T | 1549 | 1285 |
| TARBELL DISK CONTROLLER DD | 495 | 445 |
| TARBELL CASETTE INTERFACE KIT | 120 | 109 |
| SUPERBRAIN | 2995 | 2395 |
| SUPERBRAIN QUAD DENSITY | 3995 | 2995 |

| | | |
|---|---|---|
| ZENITH-HEATH Z-89 48K | 2895 | 2495 |
| INTERTUBE II SMART TERMINAL | 995 | 725 |
| ZENITH-HEATH SMART TERMINAL | 995 | 739 |
| ZENITH-HEATH WH-11 16bit COMPUTER | | 2995 |
| CAT NOVATION MODEM | 179 | 169 |
| MICROTEK PRINTER | 795 | 675 |
| ANADEX PRINTER DP-8000 | 995 | 865 |
| ANADEX PRINTER DP-9500-1 | 1650 | 1389 |
| NEC PRINTER Fast Typewriter Quality | 2915 | 2799 |
| SECRETARY WORD PROCESSOR The Best! | 85 | 77 |
| TEXTWRITER III Book Writing Program | 125 | 112 |
| GOFAST NORTH STAR BASIC Speeder Upper | 79 | 71 |
| PDS SUPER Z-80 ASSEMBLER & More | 99 | 89 |
| SUPER BASIC DEBUGGER $89 COMPILER 135 HDS | | 40 |
| EZ-80 MACHINE LANGUAGE TUTOR 25 STATISTICS | | 190 |
| EZ-CODER Translates English to BASIC | 79 | 71 |
| ECOSOFT FULL ACCOUNTING PKG | 350 | 315 |
| BOX OF DISKETTES 29 IN PLASTIC CASE | | 30 |
| Which Computers are BEST? BROCHURE.................... | | FREE |
| North Star Documentation refundable w/HRZ | | 20 |

ORDER 2 or more COMPUTERS....BIGGER DISCOUNTS
YES WE WILL BEAT OUR COMPETITION'S PRICE!
FACTORY ASSEMBLED & FACTORY WARRANTY

**AMERICAN SQUARE COMPUTERS** ✓56

**KIVETT DR • JAMESTOWN NC 27282**
**(919)-889-4577**

# What Is the Utility Of a Utility?

*A profusion of information at a low cost, for one.*

*Frank J. Derfler, Jr*
*PO Box 691*
*Herndon, VA 22070*

The unique data base held by the typical home computer user could probably be kept as easily in a loose-leaf notebook or a stack of 3 × 5 cards. Each user has disks or cassettes full of programs, but they are many of the same programs other computer users have. The amount of data unique to each individual is limited.

Economically, it makes sense to centralize the computing capability in one system, which professionals can maintain, program and interface. Individuals can subscribe to the centralized system and share both the reasonable cost and huge capabilities. Businesses and corporations have been doing this with time-sharing services for years, but time-shared large computer systems have not been economically available to individuals in the United States.

## Hobby Systems

Computer hobbyists solved the problem in their own way by establishing services using microcomputers and one-at-a-time call-in access. These are known as Computer Bulletin Board Services, Apple Bulletin Board Services or Forum-80. They generally provide a public message capability, often can transfer programs and sometimes allow running programs. But they have little or no capability for storing individual files.

Such services fill a big gap between home systems and large time-shared mainframes, but they do not have the storage or the access to large data bases unique to big computer utilities. Individuals in other countries have been using and enjoying the services a large computer can provide (see "Ultimate Consumer Computer," *Microcomputing*, October 1979, p. 94). These services are now available to you via the telephone and your own computer or home terminal.

## The Source

This system is the biggest and the best to arrive on the scene in a long time. I will not be too specific about the features of The Source, because they change so fast.

An information utility is an entry into the public data base in an interactive mode. This public data base includes information you would see in the newspapers: stock reports, business news, real estate, classifieds, travel, shopping hints and editorials. All of this information is sorted and cataloged and ready for recall according to your interests.

The Source provides both the UPI and *New York Times* news services. Real estate is a separate category, and listings, counseling and various running programs are available. The classified ads are divided into over 100 different categories, including an active bulletin board service similar to hobby systems. Travel schedules and reservations, as well as shopping clubs, wine consultation, education programs and emergency services, are available.

The *New York Times* Consumer Data Base is a separate service acting as a current-events encyclopedia with articles on file from the *N.Y. Times* and over 60 other publications. An extensive mailbox system allows the transmission of private mail between individual users. You can review, read or send messages at any time. You can also store mail and forward it to other users. A chat mode is available for users who arrange to be on the system at the same time.

The Source operates on a Prime computer system, medium-sized hardware with about a megabyte of main solid-state memory and up to 295 megabytes of disk space. The Prime computer can also be used by subscribers to run BASIC and FORTRAN programs. Many standard programs are available, and the user can insert any unique programs and create files for permanent storage.

Connection between your home or office and The Source in McLean, Virginia, is done by dialing from your telephone to data transmission carriers, who make use of their own transmission facilities and facilities leased from other providers. Several different carrier systems exist in the U.S.

Carriers connect terminals to computers and computers to computers and provide the equipment to control the flow of data. They have some of their own microwave links and lease channels on microwave, satellite and cable systems from other major carriers. They reach their customers through the local dial telephone systems and a few dedicated lines to big users.

Their activities are regulated by the FCC, ICC and Congress. They threaten and are being threatened by the post office and more traditional carriers. Despite this, they still manage to make a buck while providing more inexpensive service than the normal long-distance rate.

Telenet and Tymnet are two of these systems used by The Source. These services have over 300 entry points across the country. If you are near an entry point, you will have no long-distance toll charges.

When I lived in Montgomery, Alabama, the two nearest entry points were Birmingham and Atlanta. Birmingham was closer, but a call to Atlanta was less expensive because of the interstate rate structure. I had to pay for a long-distance call to Atlanta to use The Source, but the transmission from Atlanta to Virginia was paid for out of my user fee. I have been assured that data transmission services are going to mushroom in 1980.

A user needs a terminal and modem. The terminal can be a printer or a CRT. If it can save the information it receives for playback, retransmission or manipulation, it is a smart terminal. Smart or dumb terminals can be used on computer utilities. The modem connects the terminal to the phone line. The Bell 103/ originate modem at 300 baud is the standard for the services most of us are interested in.

This service is not free. Each user pays a one-time connect charge of $100. Billing for computer and transmission service time is divided into low and high periods. The high period occurs weekdays from 7:00 AM to 6:00 PM; the low period occurs any other time. The high-time cost is $15 per hour, and the low-period cost is $4.25 per hour for most services. Additionally, if you store data, you will pay 3.3 cents per 2048-character block per day.

The one area in which The Source could stand improvement is written documentation and explanation. The billing scheme is poorly explained. High/low billing times are determined by recorded home address, regardless of where you are calling from.

**You can buy a lot of five-dollar hours for all of the money you might spend on disk drives, memory and languages.**

Some services are never billed at the low rate. The management provides a loose-leaf binder with introductory information, but you have to spend several hours of serious study before you understand the services.

I am sympathetic to the difficulty involved in providing published documentation on a system evolving so quickly, but it certainly could be more descriptive. There are several different commands used to exit from specific services. If you don't use the correct command, you may not get out until you hang up the phone. Don't hang up without signing "OFF," or you may be billed for extra time.

The documentation is a small hurdle easily overcome with experience.

Other minor irritants are (1) the system sometimes crashes and leaves you hanging and (2) the system is often busy, particularly between 6 PM and 11 PM. I understand that computer expansion is coming, so both of these situations, common to many systems, should be improved.

The Source is a tremendous system with features previously seen only in dedicated military and educational networks. Its potential for growth is enormous, and I do not doubt that services such as this will be accessed in some form from the majority of the homes in the U.S. within seven years.

### MicroNet

MicroNet is growing so quickly that it is almost impossible to write about in a paper-and-print publication. It is a purer computer utility as contrasted to the information utility The Source claims to be. MicroNet, which is heavily targeting micro users, is a big time-sharing system used by many large government agencies and commercial corporations. It features resident software routines tailored for Apple and TRS-80 users. It does not provide the large public data base available through The Source, but it is less expensive to join. It has begun operation of a bulletin board message service and stock quotation service.

From my experience, I feel that MicroNet is more reliable than The Source. If you are mainly interested in programs and computing, MicroNet is a bargain. A one-time charge of $9 and an off-hour rate of $5 per hour allows you to use a tremendously powerful concentration of 15 mainframe computers running at least eight programming languages and a large menu of utility programs. You have to ask yourself why you would struggle to put together an extensive home computer system when that much power is available economically. You can buy a lot of five-dollar hours for all of the money you might spend on disk drives, memory and languages.

Each person must analyze his own interests and needs and buy accordingly. The spectrum from the 4K breadboard micro to the cluster of large mainframes has room for every system and user. Utilities such as Source and MicroNet add to the flexibility available to the hobbyist and small businessman. The expansion of these utilities and the carriers they use will bring an ever-increasing amount of power to the small user. If you decide to tap The Source, send a message to me at TCB967.

For information on The Source, contact Source Telecomputing Corp. (STC), 1616 Anderson Road, McLean, VA 22102.

For information on MicroNet, contact Personal Computing Division, CompuServe, Incorporated, 5000 Arlington Centre Blvd., Columbus, OH 43220. ■

In mid-May, The Source issued a new Users' Manual and Master Index which provided much better explanations and was easier to follow than the old manual.

# Darkroom Computerist

*Focus in on a "picture perfect" way to use your SWTP.*

*Marc I. Leavey, M.D.*
*4006 Winlee Road*
*Randallstown MD 21133*

I believe quite a few "computerphiles" are interested in photography. Ask a few of your friends, and I bet you uncover someone with a closet darkroom. Listen to his gripes.

What irritates me is making a test exposure each time I change enlargement ratios, and setting up title slides for the numerous slide presentations I give. This article contains programs that make both these problems vanish.

## The Enlarger

Fig. 1 is a diagrammatic representation of a common photographic enlarger. The negative is placed in a carrier in the head, and a light source projects the image onto the easel—kind of like a vertical slide projector. On the easel is a piece of photographic paper, with a light-sensitive coating that reacts to the image to produce a print, obtainable after immersing that paper in a series of chemical baths.

The size of the image is regulated by the height of the head on the column, and the focal length of the lens. If you assume a constant lens, move the head up and the image gets larger, down and the image size decreases. However, the time the paper must be exposed is a function of the amount of light falling on it. Remember the inverse-square law? The light on that easel varies as the square of the height. Add to that the iris, a variable aperture that regulates the amount of light the head puts out in the first place, and you can see how things can become complicated.

## An Example

You finish making several 4 × 6 prints of the kids, then decide to shoot for an 8 × 10. So you raise the enlarger head, cropping a little until it looks "right," and make a test exposure. From that test you derive the new exposure values, set them up and make that big print. A knock on the door introduces your spouse, who wants "just a few wallet-sized for my folks." You crank the head way down. "Gee, that's bright!" Inverse-square law, you know. Maybe one or two test exposures later, some finished "wallets."

Why the fuss? Because each of those test prints costs you time and money . . . maybe not much if you are working in black and white, but color costs lottabucks! The pros realize this, and there are gizmos costing hundreds of dollars into which you program data on an "ideal print," that is, one you consider perfect. A probe reads the light on the easel and directs you or the enlarger to set the correct time and aperture to produce a perfect print.

## Another Way

Using that old inverse-square law, there is another way to do it. If you have data on a "perfect" print, expressed as height of enlarger head above the easel ($H_O$) and time of exposure ($T_O$), the formula:

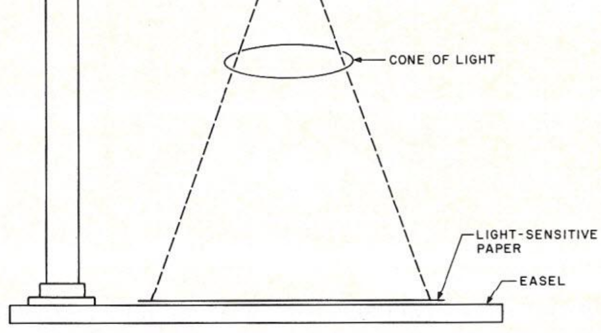



*Fig. 1. Photographic enlarger.*

```
0010 REM EXPOSURE TABLE PROGRAM
0020 REM ----------------------
0030 REM VER 2.0 -- 11 APR 1979
0040 REM  MARC I. LEAVEY, M.D.
0050 REM
0060 REM SET LINE LENGTH TO FULL WIDTH
0070 REM AND DIGITS TO FLOATING POINT
0080 LINE= 0:DIGITS=0
0090 REM BASE=0 ALLOWS SUBSCRIPT OF 0
0100 REM AS IN F(0)
0110 BASE= 0
0120 REM READ IN F-STOPS
0130 FOR I=0 TO 5
0140 READ F(I)
0150 NEXT I
0160 DATA 4,5.6,8,11,16,22
0170 HOME
0200 INPUT "WHAT KIND OF PAPER",P$
0210 PRINT "FOR A GOOD PRINT, WHAT IS:"
0220 INPUT "ENLARGER HEIGHT",S
0230 INPUT "UNITS [IN OR CM]",S$
0240 IF S$="IN" GOTO 260
0250 IF S$<>"CM" GOTO 230
0260 INPUT "EXPOSURE TIME (SEC)",T
0270 INPUT "F-STOP",F
0280 LET K=-1
0290 FOR I=0 TO 5
0300 IF F=F(I) THEN K=2↑I
0310 NEXT I
0320 IF K>-1 GOTO 400
0330 PRINT "F-STOPS ON LENS ARE:"
0340 FOR I=0 TO 5
0350 PRINT F(I)
0360 NEXT I
0370 GOTO 270
0400 PRINT "ENTER TABLE DELIMITERS:"
0410 INPUT "MINIMUM ENLARGER HEIGHT",M1
0420 INPUT "MAXIMUM ENLARGER HEIGHT",M2
0430 INPUT "INCREMENT OF HEIGHTS",I
0440 INPUT "PRINT TABLE ON WHICH PORT",P
0500 PRINT #P
0510 PRINT #P,TAB((30-LEN(P$))/2);P$
0520 PRINT #P,TAB((30-LEN(P$))/2);
0530 FOR J=1 TO LEN(P$)
0540 PRINT #P,"-";
0550 NEXT J
0560 PRINT #P
0570 PRINT #P
0580 PRINT #P,S$;
0590 FOR J=0 TO 5
0600 PRINT #P,TAB(5+5*J);F(J);
0610 NEXT J
0620 PRINT #P
0630 FOR D=M1 TO M2 STEP I
0640 PRINT #P,D;
0650 FOR N=0 TO 5
0660 LET Q=((((D/S)↑2)*T)/K)*(2↑N)
0670 IF Q<2 GOTO 710
0680 IF Q>99 GOTO 710
0690 PRINT #P,TAB(5+5*N);INT(Q+.5);
0700 GOTO 720
0710 PRINT #P,TAB(5+5*N);"-- ";
0720 NEXT N
0730 PRINT #P
0740 NEXT D
0750 SKIP #P,50-D
0760 INPUT "ANOTHER TABLE",A$
0770 IF LEFT$(A$,1)="Y" GOTO 200
0999 END
```

*Program 1.*

$$T_N = \left(\frac{H_N}{H_O}\right)^2 \times T_O$$

calculates the new time of exposure at the new height, if the same aperture is used.

Aperture is measured in f-stops, a logarithmic expression of how much light the enlarger is putting out. By reducing the expression to a common f-stop, you can produce factors that

```
WHAT KIND OF PAPER ? POLYCONTRAST - NO FILTER
FOR A GOOD PRINT, WHAT IS:
ENLARGER HEIGHT ? 20
UNITS [IN OR CM] ? IN
EXPOSURE TIME (SEC) ? 14
F-STOP ? 8
ENTER TABLE DELIMITERS:
MINIMUM ENLARGER HEIGHT ? 6
MAXIMUM ENLARGER HEIGHT ? 35
INCREMENT OF HEIGHTS ? 1
PRINT TABLE ON WHICH PORT ? 3

  POLYCONTRAST - NO FILTER
  ------------------------
```

| IN | 4 | 5.6 | 8 | 11 | 16 | 22 |
|---|---|---|---|---|---|---|
| 6 | -- | -- | -- | 3 | 5 | 10 |
| 7 | -- | -- | -- | 3 | 7 | 14 |
| 8 | -- | -- | 2 | 4 | 9 | 18 |
| 9 | -- | -- | 3 | 6 | 11 | 23 |
| 10 | -- | -- | 4 | 7 | 14 | 28 |
| 11 | -- | 2 | 4 | 8 | 17 | 34 |
| 12 | -- | 3 | 5 | 10 | 20 | 40 |
| 13 | -- | 3 | 6 | 12 | 24 | 47 |
| 14 | -- | 3 | 7 | 14 | 27 | 55 |
| 15 | -- | 4 | 8 | 16 | 32 | 63 |
| 16 | 2 | 4 | 9 | 18 | 36 | 72 |
| 17 | 3 | 5 | 10 | 20 | 40 | 81 |
| 18 | 3 | 6 | 11 | 23 | 45 | 91 |
| 19 | 3 | 6 | 13 | 25 | 51 | -- |
| 20 | 4 | 7 | 14 | 28 | 56 | -- |
| 21 | 4 | 8 | 15 | 31 | 62 | -- |
| 22 | 4 | 8 | 17 | 34 | 68 | -- |
| 23 | 5 | 9 | 19 | 37 | 74 | -- |
| 24 | 5 | 10 | 20 | 40 | 81 | -- |
| 25 | 5 | 11 | 22 | 44 | 88 | -- |
| 26 | 6 | 12 | 24 | 47 | 95 | -- |
| 27 | 6 | 13 | 26 | 51 | -- | -- |
| 28 | 7 | 14 | 27 | 55 | -- | -- |
| 29 | 7 | 15 | 29 | 59 | -- | -- |
| 30 | 8 | 16 | 32 | 63 | -- | -- |
| 31 | 8 | 17 | 34 | 67 | -- | -- |
| 32 | 9 | 18 | 36 | 72 | -- | -- |
| 33 | 10 | 19 | 38 | 76 | -- | -- |
| 34 | 10 | 20 | 40 | 81 | -- | -- |
| 35 | 11 | 21 | 43 | 86 | -- | -- |

*Sample run 1.*

allow calculation of correct time at any possible opening.

Integrating this idea into a workable scheme results in Program 1, written in SWTP BASIC. The program first specifies the type of paper used, as different photographic emulsions have different characteristics. Enter the data for a correct print: enlarger height, time for exposure and f-stop.

After requesting delimiters for a table, the program calculates the exposure time for the range of specified heights, at all available f-stops. Sample resultant tables for two different emulsions are shown in Sample runs 1 and 2. Different types of paper—one black and white, one color—illustrate the different requirements varied emulsions produce.

Note that times greater than 99 seconds or less than two seconds are not printed; a dash is inserted instead. This is in deference to most timers on the market today. Also, while printed times are rounded off to the nearest whole second, the actual values are used in calculations to maintain accuracy.

### How to Use This Program

Make a good exposure of an average negative. Feed the requested data into the computer, then take the table printed out to the darkroom. When you change enlargement ratios, simply measure the enlarger height and read the appropriate exposure time and f-stop off the table. An accurate, program-

mable timer, such as the one described in *73 Magazine*, August 1977, pp. 66–71 ("Build a Unique Timer," WA3AJR), is an aid to setting repeatable times for this program.

### A Finishing Touch

You can use Program 2 to identify slides when you project them. This program is set up for a scrolling terminal, with H set to the number of characters per line and V set to the number of lines. It inputs up to three less than V lines, centers them and surrounds them with a border character of your choosing. If you don't want a border, you can use a nonprinting single character such as CONTROL-G (BELL). A loop at the end ties things up until you take a picture of the screen.

I put titles on the screen and shoot them, after thoroughly cleaning the screen, with ASA 125 Plus-X at f/1.4, 1/8 second. I use a tripod and darken the room to eliminate reflections from the screen. The resultant negatives are a black-on-clear representation of my white-on-black background display and make snazzy title slides. You can mount them in slide mounts alone or, with a little creative planning, shoot them to sandwich with a color slide, producing a superimposed title. ■

```
0010 REM  SLIDE TITLE MAKER
0020 REM -------------------
0030 REM VER 2.0 - 13 APR 79
0040 REM MARC I. LEAVEY, M.D.
0050 REM
0060 REM SET LINE LENGTH
0070 LINE= 0
0080 REM "H" = CHARACTER WIDTH OF SCREEN
0090 REM "V" = NUMBER OF LINES ON SCREEN
0100 LET H=32
0110 LET V=16
0120 DIM L$(V),L(V)
0130 INPUT "BORDER CHARACTER",B$
0140 IF LEN(B$)=1 GOTO 200
0150 PRINT "PLEASE TYPE ONLY ONE CHARACTER"
0160 GOTO 130
0200 LET K=0
0210 FOR N=1 TO V-3
0220 PRINT "LINE #";N;
0230 INPUT L$(N)
0240 IF LEN(L$(N))<=H-5 GOTO 300
0250 PRINT CHR$(7);"LINE TOO LONG!";CHR$(7)
0260 GOTO 220
0300 IF L$(N)="#$" THEN N=N-1:GOTO 400
0310 LET L(N)=LEN(L$(N))
0320 IF L(N)>K THEN K=L(N)
0330 NEXT N
0400 LET I=K+4
0410 LET J=((H+1)-I)/2
0420 PRINT TAB(J);
0430 FOR M=1 TO I
0440 PRINT B$;
0450 NEXT M
0460 PRINT
0470 FOR M=1 TO N
0480 PRINT TAB(J);B$;
0490 PRINT TAB(((H+1)-L(M))/2);L$(M);
0500 PRINT TAB(I+J-1);B$
0510 NEXT M
0520 PRINT TAB(J);
0530 FOR M=1 TO I
0540 PRINT B$;
0550 NEXT M
0560 FOR M=1 TO V-(N+2)
0570 PRINT
0580 NEXT M
0590 REM LOOP TO KEEP THINGS BUSY
0600 GOTO 590
```

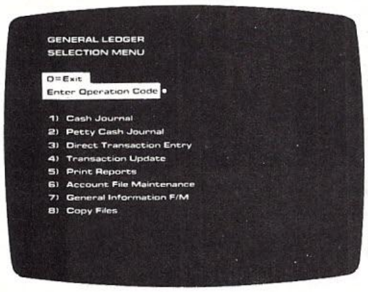
*Program 2.*

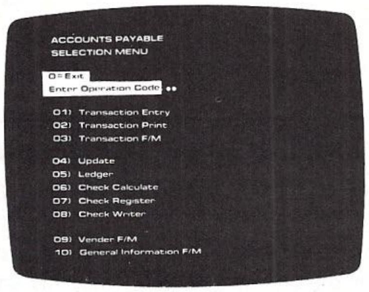
```
WHAT KIND OF PAPER ? CIBACHROME
FOR A GOOD PRINT, WHAT IS:
ENLARGER HEIGHT ? 21
UNITS [IN OR CM] ? IN
EXPOSURE TIME (SEC) ? 10
F-STOP ? 5
F-STOPS ON LENS ARE:
4
5.6
8
11
16
22
F-STOP ? 4
ENTER TABLE DELIMITERS:
MINIMUM ENLARGER HEIGHT ? 10
MAXIMUM ENLARGER HEIGHT ? 24
INCREMENT OF HEIGHTS ? 1
PRINT TABLE ON WHICH PORT ? 3

          CIBACHROME
          ----------

IN   4    5.6   8    11   16   22
10   2    5     9    18   36   73
11   3    5     11   22   44   88
12   3    7     13   26   52   --
13   4    8     15   31   61   --
14   4    9     18   36   71   --
15   5    10    20   41   82   --
16   6    12    23   46   93   --
17   7    13    26   52   --   --
18   7    15    29   59   --   --
19   8    16    33   65   --   --
20   9    18    36   73   --   --
21   10   20    40   80   --   --
22   11   22    44   88   --   --
23   12   24    48   96   --   --
24   13   26    52   --   --   --
```

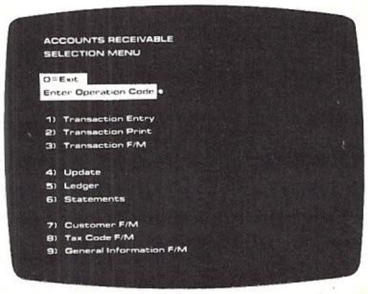
*Sample run 2.*

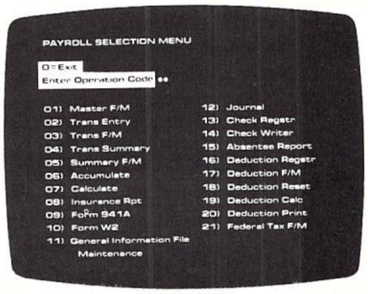
# Professional Business Software

## For The Commodore 32K Microcomputer System
## With 2040 Dual Drive Disk & 2022 Tractor Feed Printer




### General Ledger

- Holds Up To 300 Accounts.
- Accepts Up To 3000 Transactions Per Month.
- Cash Disbursements Journal, Cash Receipts Journal, and Petty Cash Journal for simplified data entry.
- Maintains Account Balances For Present Month, Present Quarter, Present Year, Three Previous Quarters, And Previous Year.
- Complete Financial Reports Including Trial Balance, Balance Sheet, Profit & Loss Statement, Cash Receipts Journal, Cash Disbursements Journal, Petty Cash Journal and more.
- Accepts Postings From External Sources Such As Accounts Payable, Accounts Receivable, Payroll, Etc. . . . . . . . . . . . . . . . **$295.00**

### Accounts Payable

- Interactive Data Entry With Verified Input And Complete Operator Prompting.
- Automatic Application Of Credit And Debit Memos.
- Maintains Complete Purchase Records For Up To 200 Vendors.
- Invoice File Accepts Up To 400 Invoices.
- Random Access File Organization Allows Fast Individual Record Updating.
- Multiple Reports Provide A Complete Audit Trail.
- Check Printing With Full Invoice Detail.
- Full Invoice Aging.
- Automatic Posting To General Ledger . . . .**$195.00**

### Accounts Receivable

- Maintains Invoice File For Up To 300 Invoices.
- Accomodates Full Or Partial Invoice Payments.
- Customer File Maintains Purchase Information For Up To 1000 Customers.
- Allows For Automatic Progress Billing.
- Provides For Credit And Debit Memos As Well As Invoices.
- Prints Individualized Customer Statements.
- Interactive Data Entry With FullOperator Prompting.
- Complete Data Input Verification And Formating.
- Automatic Posting To General Ledger . . . .**$195.00**

### Payroll

- Maintains Monthly, Quarterly, And Yearly Cumulative Totals For Each Employee.
- Payroll Check Printing With Full Deduction And Pay Detail.
- Sixteen Different Reports Including W2 And 941.
- Interactive Data Entry With Easy Correction Of Entry Errors.
- Automatic Data Verification.
- Complete Job Costing Option With Cumulative Totals And Overhead Calculations.
- Random Access File Organization For Fast Updating Of Individual Records.
- Automatic Posting To General Ledger . . . . $350.00

---

Structured around the time tested and reliability proven series of business software systems developed by Osborne and Associates, these programs have been designed to fill the need of a comprehensive accounting package for the new Commodore PET micro computer system. Each program can either stand alone, or be integrated with the others in a total software system.

Designed with the first time user in mind, these programs lead the operator through step by step, verified data entry. It is impossible to 'crash' a program due to operator error or invalid data input. Design consistency has been maintained from program to program to greatly increase operator familiarity and confidence.

Documentation, normally a problem for small systems users, is provided by the comprehensive series of Osborne and Associates user manuals. These three manuals together total over 800 pages of detailed step by step instructions written at three levels for DP Department Managers, Data Entry Operators, and Programmers. You don't have to worry about getting 'promises' instead of documentation because the documentation was written before the programs were developed. A second set of manuals details any changes required during conversion. Each program provided on disk with complete documentation. Packaged in a handsome three ring binder with pockets and twelve monthly dividers for convenient storage of reports.

***See your nearest Commodore dealer for a demonstration of this outstanding business software system.***

**CMS Software Systems** ✓5

5115 MENEFEE DRIVE • DALLAS, TX 75227 • 214-381-0690

# Start/Exit Routine for CP/M

*This assembly-language utility provides links between CP/M and user programs and permits simplified operation of remote terminals.*

*Ken Barbier*
*Borrego Engineering*
*PO Box 1253*
*Borrego Springs, CA 92004*

If you program in assembly language under CP/M, you might find that Start/Exit routine in this article listing to be a useful addition to your program library. While basically written to provide orderly linkages between CP/M and user programs under console control, this program provides other convenient features, all in less than 256 bytes:

1. It allows you to quickly copy .COM files from disk to disk, even if you have only a single disk drive.
2. It provides an exit from CP/M to your built-in ROM monitor, or to other bootstrap programs in ROM.
3. It provides a return to CP/M from user programs or from a ROM monitor.
4. By allowing you to jump between CP/M, user programs and the ROM monitor under console control, it permits the operation of remote terminals without the need for access to the computer's RESET switch or front panel.
5. It includes a console message output subroutine that permits you to embed console messages anywhere within your assembly-language programs, much like BASIC PRINT statements.

### The Start Program

The source program resides on disk as library file START.LIB and should be appended at the beginning of any assembly-language source program by entering the ED statement "RSTART." When thus used as a library file, line 79 should be replaced by "ORG 0200H", so that your user program will now start at location 200 instead of 100, where Start must reside at the beginning of the transient program area.

When your assembled user program (including Start) that is on the disk as a .COM file is called by name, CP/M will load your program and jump to Start, which will then display the following console message:

B = TAPE BOOTSTRAP
C = CP/M RESTART
M = MONITOR
R = RUN PROGRAM
?:

where ?: is prompting you for one of the four single-letter directive choices. If you enter the wrong letter, hit the Rubout key before a carriage return, and Start will reprompt with ?:.

When the correct choice is entered, followed by a carriage return, Start will jump to the desired program. If your choice is R to run your user program, then that program can return to Start at address 100 after it has done its thing, allowing you to make any of the four choices as before.

If you have a good monitor program in ROM, you might prefer to use it instead of CP/M's DDT to debug your new user program, so you would enter M instead of R. When finished with the monitor, it can reenter Start at location 100 to provide a link back to CP/M.

### The Exit Program

If you assemble this program with line 79 as an END statement as shown in the listing and name the object file EXIT.COM, then this same routine can be used as an exit from CP/M to your monitor or a tape bootstrap program. This is especially useful if CP/M is a recent addition to your system, and you have mountains of software on cassette tapes.

I hope that when you ran those cables from the computer room to the bedroom (so you could play in comfort), you included a couple of audio cables so you could take your cassette along with you. Now you can easily switch between disk programs under CP/M and cassette programs at your remote terminal without having to get up to hit the reset switch.

### A Convenient Message Subroutine

Start/Exit includes a console message subroutine that allows you to place console messages anywhere within your program and output them to the console with a minimum of overhead. For example, you can output an error message with this simple sequence:

```
CALL  MSGXP
DB    'ERROR!'
DB    0DH,0AH,0
```

Your message can be any length, and the length parameter does not have to be passed to MSGXP. Your message text must terminate in a zero, and the subroutine will return to your calling program at the state-

*Program listing. Start/Exit program in assembly language.*

```
 8:  3E09 =              CI      EQU   3E09H    ; BIOS 'CONIN'
 9:  3E0C =              CO      EQU   3E0CH    ; BIOS 'CONOUT'
10:  3E03 =              WBOOT   EQU   3E03H    ; WARM START
11:  2900 =              STAK    EQU   2900H    ; TOP OF USEABLE RAM
12:  C800 =              MONITOR EQU   0C800H   ; ROM MONITOR
13:  C803 =              TBOOT   EQU   0C803H   ; ROM TAPE BOOTSTRAP
14:  0200 =              PROGRAM EQU   0200H    ; PROGRAM START
15:
16:  0100                        ORG   0100H             ; TPA START
17:
18:  0100 310029         START   LXI   SP,STAK           ; SETUP STACK
19:  0103 FB                     EI                      ; ENABLE INTS
20:  0104 CD9101                 CALL  MSGXP             ; SIGN - ON
21:  0107 0D0A                   DB    0DH,0AH           ; CR, LF
22:  0109 42203D2054             DB    'B = TAPE BOOTSTRAP'
```

```
011B 0D0A                DB      0DH,0AH
011D 43203D2043          DB      'C = CP/M RESTART'
012D 0D0A                DB      0DH,0AH
012F 4D203D204D          DB      'M = MONITOR'
013A 0D0A                DB      0DH,0AH
013C 52203D2052          DB      'R = RUN PROGRAM'
014B 0D0A00              DB      0DH,0AH,0

014E CD9101      STAR0   CALL    MSGXP                   ; PROMPT INPUT
0151 0D0A                DB      0DH,0AH
0153 3F3A20              DB      '?: '
0156 00                  DB      0

0157 CD093E      STAR1   CALL    CI                      ; GET DIRECTIVE
015A F5                  PUSH    PSW                     ; SAVE IT
015B 4F                  MOV     C,A                     ; ECHO IT
015C CD0C3E              CALL    CO
015F CD093E      STAR2   CALL    CI                      ; WAIT FOR CR
0162 F5                  PUSH    PSW
0163 4F                  MOV     C,A
0164 CD0C3E              CALL    CO
0167 F1                  POP     PSW
0168 FE0D                CPI     0DH                     ; IS INPUT CR?
016A CA7501              JZ      STAR3
016D FEFF                CPI     0FFH                    ; IS IT RUBOUT?
016F CA8D01              JZ      STAR4                   ; YES, RESTART
0172 C35F01              JMP     STAR2                   ; GET NEXT CHAR
0175 F1          STAR3   POP     PSW                     ; RESTORE CHAR
0176 FE42                CPI     'B'                     ; PROCESS DIREC
0178 CA03C8              JZ      TBOOT
017B FE43                CPI     'C'
017D CA033E              JZ      WBOOT
0180 FE4D                CPI     'M'
0182 CA00C8              JZ      MONITOR
0185 FE52                CPI     'R'
0187 CA0002              JZ      PROGRAM
018A C34E01              JMP     STAR0                   ; ILLEGAL!
018D F1          STAR4   POP     PSW                     ; KILL DIREC
018E C34E01              JMP     STAR0                   ; GET NEW ONE

                 *
                 * OUTPUT MESSAGE TO CONSOLE
                 *    AND RETURN THRU INDEX
                 *

0191 E1          MSGXP   POP     H           ; GET CALL ADRS + 1
0192 7E          MSGX1   MOV     A,M         ; GET CHARACTER
0193 FE00                CPI     0           ; TIL END OF MESSAGE
0195 CAA001              JZ      MSGEX       ; THEN EXIT
0198 4E                  MOV     C,M         ; CHARACTER TO (C)
0199 CD0C3E              CALL    CO          ; OUTPUT CHARACTER
019C 23                  INX     H           ; AHEAD ONE
019D C39201              JMP     MSGX1       ; CONTINUE TIL DONE
01A0 23          MSGEX   INX     H           ; AHEAD TO NEXT
01A1 E9                  PCHL                ; INSTRUCTION, AND GO!

01A2                     END
```

ment immediately following the zero.

MSGXP uses the A, C and H, L registers, so if you need to preserve the contents of these registers, you will have to surround your CALL and the message text with appropriate PUSH and POP statements.

There is quite a bit of program space left between the top of Start and the beginning of user program space at location 200. You can use this room for additional jump options. Both the header message text and the directive processing in the program listing are organized to make it easy to add options.

In any case, it would be a good idea to leave the start address of all user programs at location 200, even if some space is wasted. Then you will always know where your program begins.

### Single Drive Disk Copy

With Start appended to your program, it is easy to copy that program onto another CP/M disk even with a single drive system. When Start prompts with ?: place the destination disk in the drive and enter C and a carriage return to reboot CP/M from the new disk. When CP/M prompts with "A>" enter "SAVE x X.COM", where X is your program name and x is the appropriate size in 256 byte blocks.

By rebooting CP/M, you enable writing onto the new disk. Your user program remains undisturbed in memory during the reboot, ready to be saved.

The absolute addresses in the EQUate statements are for a 16K version of CP/M. The addresses given for the monitor and tape bootstrap programs are for a specific hardware system, and you will have to change them to the proper entry points for your own ROM programs.■

# Modifying the Horizon Double Density DOS

*Programming tricks to really personalize your system.*

*George L. Haller*
*1500 Galleon Drive*
*Naples, FL 33940*

The Horizon disk operating system for double density is a wonderful collection of instructions. But it lacks some of the personalized features that would make it even better.

**The Modifications**

There are three modifications I think can improve the system.

1. On many occasions, I would like to have the video display and the printer work at the same time without modifying the BASIC program. Since the first serial output is for the video terminal and the second serial output is available for a peripheral, all I need is to make them work together in tandem. I then will have the video displaying the output while the output is also being printed.

2. I would like the ability to output the desired number of nulls. Some peripherals in the serial mode require a certain number of nulls after CR/LF. This feature was included as a command in the older North Star BASIC, but was eliminated some versions ago.

3. I would like to be able to output spaces after a CR/LF. This feature is useful when a line printer prints too close to the left-hand edge of the paper and does not leave enough space for binding the pages. It is especially convenient for listing where you do not have control of the page format.

The first modification is simple because of the output code setup. The first serial output is a subroutine starting at 293EH with a Return, C9H, at 2948H. This is followed by the second serial output at 2949H with a Return at 2953H. All you need to link the two outputs is to replace the first Return with a NOP, and you will be able to watch the video while the printer is working away.

This is easily done in BASIC by Fill 10568,0. To return to normal operation use Fill 10568,201. 201 is the decimal equivalent of C9H, which is Return. This is shown in Listing 1.

The second modification, to add nulls, is more difficult, since you must key on the CR/LF to determine when to add the nulls. Actually, you key on the line-feed character. Listing 2 shows the necessary changes. There is adequate space beginning at 29BCH to accommodate your new sub program so that at COUT1, 2949H, you will jump to the patch named COUT3 with C3,BC,29.

The remainder of the codes in COUT1 are not used. The codes shown in Listing 1 are inserted and used. This particular set inserts seven nulls, but this number may be changed at 29CCH.

The third modification, the insertion of spaces at the beginning of each line, can be assembled similarly. Listing 3 shows a combination of both nulls and spaces, with seven nulls and five spaces.

These programs are versatile, and if residing in DOS, may be changed by the BASIC FILL command.

```
                  ;THE FOLLOWING IS A PATCH IN USER'S AREA OF N*
                  ;HORIZON DOUBLE DENSITY DOS TO ALLOW THE
                  ;OUTPUT ON TERMINAL 1 AND TERMINAL 2 IN
                  ;TANDEM AND CONTINUOUSLY. THIS PATCH IS
                  ;IS MERELY CHANGING 2948 FROM C9 TO NOP.
                  ;THIS CAN BE DONE IN BASIC BY FILL 10568,0
                  ;AND RETURNED TO NORMAL BY FILL 10568,201


2933E DB03      COUO    IN      3       ;INPUT FIRST SERIAL PORT STATUS
2940 E601               ANI     1       ;MASK OUTPUT STATUS BIT
2942 CA3E29             JZ      COUO    ;LOOP IF NOT READY TO OUTPUT
2945 78                 MOV     A,B     ;MOVE CHARACTER TO A
2946 D302               OUT     2       ;OUTPUT THE CHARACTER
2948 C9                 RET             ;CHANGE TO NOP TO OUTPUT
                                        ;  TO BOTH SERIAL PORTS
2949 DB05       COUT1   IN      5       ;INPUT SECOND SERIAL PORT STATUS
294B E601               ANI     1       ;MASK INPUT STATUS BIT
294D CA4929             JZ      COUT1   ;LOOP IF NOT READY TO OUTPUT
2950 78                 MOV     A,B     ;MOVE CHARACTER TO A
2951 D304               OUT     4       ;OUTPUT THE CHARACTER
2953 C9                 RET
```

*Listing 1. Modification to link two outputs.*

```
10 Input "DEVICE NUMBER",Z
20 PRINT #Z, "THIS WILL PRINT ON THE SELECTED DEVICE"
30 PRINT "THIS WILL PRINT ON THE VIDEO TERMINAL"
```

*Table 1.*

```
                 ;THE FOLLOWING IS A PATCH IN USER'S AREA OF N*
                 ;HORIZON DOUBLE DENSITY DOS TO INSERT NULLS AND
                 ;SPACES AFTER A CR/LF. THIS PATCH IS JUMPED TO BY THE
                 ;SMALL PATCH FROM THE BEGINNING OF COUTI AT 2949H.
                 ;01/10/80

2949 C3BC29

29BC 78          COUT3   MOV    A,B      ;PUT CHAR IN REG A
29BD FEOA                CPI    OAH      ;IS IT A LF?
29BF CAC529              JZ     LNFD     ;YES
29C2 C3E829              JMP    OUTPT    ;NO
29C5 CDE829      LNFD    CALL   OUTPT    ;OUTPUT THE LF CHAR.
29C8 C5                  PUSH   B        ;SAVE C
29C9 0600                MVI    B,0      ;NULL CHAR. TO B REG.
29CB 0E07                MVI    C,7      ;NO. OF NULLS TO C REG.
29CD OD          NULLS   DCR    C        ;DECR. C REG.
29CE FAD729              JM     SPACE    ;IF ALL DONE W/NULLS
29D1 CDE829              CALL   OUTPT    ;OUTPUT A NULL
29D4 C3CD29              JMP    NULLS    ;DO IT AGAIN
29D7 0620        SPACE   MVI    C,5      ;SPACE CHAR. TO B REG.
29D9 0E05                MVI    C,5      ;NO. OF SPACES TO C
29DB OD          SPACE1  DCR    C        ;DECR. C REG.
29DC FAE529              JM     POP1     ;IF ALL DONE WITH SPACES
29DF CDE829              CALL   OUTPT    ;OUTPUT A SPACE
29E2 C3DB29              JMP    SPACE1   ;DO IT AGAIN
29E5 C1          POP1    POP    B        ;RESTORE C
29E6 78                  MOV    A,B      ;MAKE A=B
29E7 C9                  RET
29E8 DB05        OUTPT   IN     5        ;FOLLOWING IS A COPY
29EA E601                ANI    1        ;OF THE COUTI CODE
29EC CAE829              JZ     OUTPT
29EF 78                  MOV    A,B
29F0 D304                OUT    4
29F2 C9                  RET
```

*Listing 3. Modification to insert spaces at the beginning of each line.*

If only the null change is made as shown in Listing 2, any number of nulls may be inserted in the DOS by FILL 10700,XX, where XX is the number of desired nulls in decimal. 10700D = 29CCH.

If the null and space version (Listing 3) is in use, then FILL 10700,XX will insert XX nulls and FILL 10714,YY will insert YY spaces.

**Implementation**

Bring up your DOS and load it at 4000H by LF 4000. Then go to the monitor at 0 or 2D00H. The monitor should be loaded where it will not interfere with the operating DOS or the loaded DOS. The loaded DOS uses 4000H to 4CFFH. The modifications may be inserted into the loaded DOS by using the FM or the DS command from the monitor. But, while the code will end up in the 2900s, it should be loaded into the 4800s instead of the 4900s.

In other words, the first three codes inserted would start at 4848H and would be C3,BC,29, and the next inserted codes would start at 48BCH and be 78,FE,0A. When you have made the modifications, use the monitor command OS to get back to the operating DOS and then SF DOS 4000, which will save your new DOS and let you try it.

One other trick in manipulating the output to the various terminals is to use the output device capability of North Star BASIC. You can put a line in the BASIC program requesting the input of the output device number, and then use it as needed in the program. While this is described in the manual, the short program in Table 1 may give one an idea of how it works.

This comes in handy when only part of the output is to be printed and the remainder goes to the video terminal. ■

```
                 ;THE FOLLOWING IS A PATCH IN USER'S AREA OF N*
                 ;HORIZON DOUBLE DENSITY DOS TO INSERT NULLS AFTER
                 ;A CR/LF. THIS PATCH IS JUMPED TO BY THE SMALL
                 ;PATCH FROM THE BEGINNING OF COUTI AT 2949H.
                 ;01/10/80

2949 C3BC29

29BC 78          COUT3   MOV    A,B      ;PUT CHAR IN REG A
29BD FEOA                CPI    OAH      ;IS IT A LF?
29BF CAC529              JZ     LNFD     ;YES
29C2 C3DA29              JMP    OUTPT    ;NO
29C5 CDDA29      LNFD    CALL   OUTPT    ;OUTPUT THE LF CHAR.
29C8 C5                  PUSH   B        ;SAVE C
29C9 0600                MVI    B,0      ;NULL CHAR TO B REG.
29CB 0E07                MVI    C,7      ;7 NULLS TO C REG.
29CD OD          NULLS   DCR    C        ;DEC. C REG.
29CE FAD729              JM     POP1     ;IF ALL DONE WITH NULLS
                                         ;RESTORE C
29D1 CDDA29              CALL   OUTPT    ;OUTPUT A NULL
29D4 C3CD29              JMP    NULLS    ;DO IT AGAIN
29D7 C1          POP1    POP    C        ;RESTORE C
29D8 78                  MOV    A,B      ;MOVE B TO A
29D9 C9                  RET
29DA DB05        OUTPUT  IN     5        ;FOLLOWING IS A COPY
29DC E601                ANI    1        ;OF THE COUTI CODE
29DE CADA29              JZ     OUTPT    ;
29E1 78                  MOV    A,B      ;
29E2 D304                OUT    4        ;
29E4 C9                  RET             ;
```

*Listing 2. Modification to add nulls.*

# DR. DALEY'S *BEST* Mailing List Is Now Better!

DR. DALEY has taken his best selling mailing list and made it even better! This version has been totally revised to increase the reliability of the files and make it even easier to operate. Several new features have been added:

- Goof-proof input routine. Eliminates the irritating results of accidentally pressing some cursor control keys. This is a machine code routine so it is as fast as you are! BONUS—Auto repeat on all keys!
- Interface to allow output of the entire mailing list or virtually *ANY* subset to WORDPRO III and WORDPRO IV format files so you can use these to generate personalized form letters. *YOU* can format the structure of this output!
- Routines to merge files and to minimize the number of duplicate entries in a file.
- More machine code routines to speed up processing.
- In addition you have the same powerful file formatting options where *YOU* can determine the structure of the files. *YOU* can format your label output with up to 11 lines per label and from 1 to 8 (yes EIGHT) labels per line.

This system is completely menu driven. It includes 100 pages of user documentation. This documentation is for the end user and is not padded with listings, flow charts, and other such extraneous material.

This program will be available for a short time at the introductory price of $159.95. It is available for the 32K PET and CBM 2000, 3000 and 8000 series computers. You can order through your dealer or directly from us. We will accept VISA or MASTERCARD or your check or money order. Overseas orders include 10% to cover shipping.

Charge to
your
MC/VISA

## DR. DALEY'S Software ✓34

**425 Grove Avenue, Berrien Springs, MI 49103**
**Phone (616) 471-5514**
**Sunday - Thursday noon to 9 p.m., Eastern Time**

✓ Reader Service index—page 241

# PET Mini Monitor

*This short routine hides in the second cassette buffer and makes creating and saving machine-language programs a snap.*

*William H. Perdue*
*2815 Pulaski Pike*
*Huntsville, AL 35810*

If you have ever tried any real-time graphics on your PET, you have undoubtedly found that BASIC is sometimes too slow. Normally, the answer is to use machine-language subroutines that can be linked to your main BASIC program through the SYS command.

However, entering the machine code for the subroutine into the machine is a clumsy and time-consuming operation, even though this can be done using BASIC. Everything must be converted from hexadecimal to decimal and then poked into the proper memory locations. Another drawback occurs when you have to locate and correct errors, or make changes, in your machine-language routines. Again, it is possible to do this using BASIC, but not very efficiently.

Another drawback becomes apparent after you get your program written, entered into the machine, with everything working correctly together, and then you try to save it on tape. The normal SAVE operation will only save the program that is written in BASIC, and the machine-language routines are ignored. The only way around this is to rewrite the entire program, putting the machine code into DATA statements and including a routine to read and poke the data into the necessary memory locations. Although this will now allow you to save the complete program in BASIC, it does use up a considerable amount of memory.

You can deal with all of these drawbacks using BASIC, although inconveniently, but there is a better way. This article describes a machine-language monitor program that allows you to read and write memory directly in hexadecimal form. This eliminates the need to convert from hexadecimal to decimal for entry, thereby reducing the possibility of errors. You can write, enter and check your programs in hex, which simplifies the task of machine-language programming. Most important is the ability to save the program once you have it written and checked out.

Machine-language monitor programs are not new; there is an excellent monitor program available for the PET that does all of the things I have covered and more. However, the program did not meet my particular needs. It required about 875 bytes of user RAM, which I was not willing to give up just for the convenience of writing machine-language code.

I needed a short and simple program that would do the job without using my BASIC text memory. Since I don't have a second cassette unit on my PET, I have 198 bytes of memory available at the end of page two of memory. With this in mind, I developed the PET Mini Monitor.

### Program Description

Mini Monitor is a machine-language program that resides in the second tape buffer. Operation of the program is similar to the KIM operating system. The program allows the user to open and display all memory addresses, modify data in RAM-type memory and save programs on tape from any point in memory. The program does not affect the normal BASIC interpreter operating system. This allows the user to access one—either the Mini Monitor or the BASIC interpreter operating system—without having any detrimental effect on the other.

### Instructions

First, you must enter the program into your machine. If you have the PET TIM program, you can use it to enter the machine code from the listing; otherwise, use the BASIC program in Listing 1. Carefully type in the program and check the DATA statements to be sure you have made no mistakes before you run the program. Once you have run the BASIC program and have the machine code in the machine, the normal READY indication should display on your screen.

If you have a new PET or have installed the new ROM set in your machine, do not try to run Mini Monitor. The zero-page allocation is different in the new operating system, and the program can cause you to lose your operating system. If you have one of the old PETs and have the normal READY indication after loading the machine code, you can run the program. You must type SYS(826) and press the return key. This will link in the machine-language program and produce the following display:

```
MM
0400 00 ( )← Blinking cursor
```

This start display should appear every time you enter the program. The MM indicates that you are running Mini Monitor and have successfully entered the program. The four-digit number is the hexadecimal memory address, which is followed by a two-digit number that shows the hexadecimal contents of the memory cell. The actual value of the two-digit number will depend on the contents of memory at the time of dis-

| | | | |
|---|---|---|---|
| 0401 | 00 | 4C | 4C entered into address 0401 |
| 0402 | 00 | 43A | 3A entered into address 0402 |
| 0403 | 00 | 3 | 03 entered into address 0403 |
| 0404 | 00 | 0 | 00 entered into address 0404 |

*Note: The actual value displayed in this column will depend on the contents of memory at the time of the display and may not be the values shown.

*Table 1.*

| | | | |
|---|---|---|---|
| 0401 | 4C | F6B1: | new address entered and RETURN pressed |
| F6B1 | A5 | ( )-- Blinking cursor | new address displayed |
| F6B1 | A5 | F1: | new address 00F1 (leading zeros assumed) |
| 00F1 | 01 | ( )-- Blinking cursor | |
| 00F1 | 01 | F6033A: | new address 033A (four characters preceding the colon) |
| 033A | D8 | ( )-- Blinking cursor | |
| 033A | D8 | : | return to start |
| MM | | | |
| 0400 | 00 | ( )-- Blinking cursor | |

*Table 2.*

| | | | |
|---|---|---|---|
| Starting Address | 04 00 Hi Lo | 05 3A Hi Lo | Ending Address |

SYS(826) RETURN pressed (enter Mini Monitor program)

| | | | |
|---|---|---|---|
| MM | | | |
| 0400 | XX* | E5: | first address to be inspected (address 00E5) |
| 00E5 | XX | 3A | enter ending address Lo byte |
| 00E6 | XX | 05 | enter ending address Hi byte |
| 00E7 | XX | F1: | next address to be inspected (address 00F1) |
| 00F1 | XX | 01 | enter 01 always (01 = Tape #1) |
| 00F2 | XX | F7: | next address to be inspected (address 00F7) |
| 00F7 | XX | 00 | enter starting address Lo byte |
| 00F8 | XX | 04 | enter starting address Hi byte |
| 00F9 | XX | 20 | always enter 20 at this address |
| 00FA | XX | 00 | always enter 00 at this address |

*Note: The XX represents the contents of the address before modification.

*Table 3.*

| Address | |
|---|---|
| 00E5 | enter FF |
| 00E6 | enter 03 |
| 00F1 | enter 01 |
| 00F7 | enter 3A |
| 00F8 | enter 03 |
| 00F9 | enter 20 |
| 00FA | enter 00 |

Type S and press RETURN
Type MINI MONITOR and press RETURN

*Table 4.*

play. To open and display the next sequential memory cell, just press the RETURN key. Your display should now appear as follows:

```
MM
0400 00
0401 00 ( )-- Blinking cursor
```

To change the data in a memory address, type the new hexadecimal data and press the return key. The program will enter the data in the last open address and then open and display the next sequential memory address. Only the last two numbers typed before the return key will be entered into memory. If only one number is typed before the return key, the most significant digit of the data is automatically assumed to be zero. Table 1 shows some examples of how to change data in a memory address.

If you want to check if the correct value was entered into memory, type ↑ (up arrow) and press the return key. This will cause the program to open and display the previous memory cell. Repeating the operation will again cause the next previous memory cell to be displayed. You can continue in this manner as long as you wish, going backward through memory. With the examples in Table 1, the display is as follows:

```
0405 00 ↑
0404 00 ↑
0403 03 ↑
0402 3A ↑
0401 4C ( )-- Blinking cursor
```

The Mini Monitor program automatically starts to open and display memory at hexadecimal address 0400 and continues sequentially after that. To change the address to be displayed next, type the hexadecimal address you wish to display, then type : (colon) and press the RETURN key. The new address followed by its contents will then be displayed. Pressing the return key will cause the next sequential address following the new address to be opened and displayed.

Only the last four characters preceding the colon will be used to establish the new address. If there are less than four characters preceding the colon, leading zeros are automatically assumed for the new address. If there are no characters preceding the colon, the program automatically returns to the start and displays address 0400. Examples of how to change the address being displayed are shown in Table 2.

When you want to return to the BASIC interpreter operating system, type X and press the return key. You should then see the familiar READY display. If not, you must turn off the power to reset the machine.

```
MM
0400 00 X    exit to BASIC
READY.
( )-- Blinking cursor
```

The Mini Monitor program allows you to name and save programs from any point in memory. However, due to the save routine utilized, certain addresses must be set to specific values to assure correct save operation. Furthermore, you must know the hexadecimal starting and ending addresses of the program you want to save. The Mini Monitor program may be used to inspect the specified addresses and modify the data as needed. These modifications must be made prior to executing the save. Table 3 lists the steps necessary to save a program that has a starting address of 0400 and an ending address of 053A. Using the Mini Monitor, you would inspect and modify

```
10 FOR AD=826 TO 1023
20 READ D$
30 L=ASC(LEFT$(D$,1))
40 R=ASC(RIGHT$(D$,1))
50 IF L>64 THEN L=L-7
60 L=(L-48)*16
70 IF R>64 THEN R=R-7
80 D=L+(R-48)
90 POKE AD,D
100 NEXT AD:END
110 DATA D8,20,36,E2,A9,4D,20,A7,03,A0,00,A9,04,85,5C,20
120 DATA D2,C9,A5,5C,20,AD,03,98,20,AD,03,20,A5,03,B1,5B
130 DATA 20,AD,03,20,E6,03,E0,00,F0,2F,B5,1F,C9,5E,D0,0A
140 DATA 88,C0,FF,D0,DA,C6,5C,4C,49,03,C9,3A,D0,0E,CA,8A
150 DATA F0,BE,20,C6,03,A8,20,C3,03,4C,47,03,C9,58,F0,5B
160 DATA C9,53,F0,0D,20,C6,03,91,5B,C8,D0,B3,E6,5C,4C,49
170 DATA 03,20,E6,03,86,EE,A2,00,4C,B1,F6,A9,20,20,D2,FF
180 DATA 4C,D2,FF,48,4A,4A,4A,4A,20,B8,03,68,29,0F,18,C9
190 DATA 0A,90,02,69,06,69,30,D0,E7,8A,F0,12,20,D9,03,E0
200 DATA 00,F0,0B,85,FE,20,D9,03,0A,0A,0A,0A,05,FE,60,CA
210 DATA B5,20,18,C9,41,90,02,69,08,29,0F,60,A2,00,20,A5
220 DATA 03,20,CF,FF,C9,0D,F0,F3,E0,10,F0,F5,95,20,E8,D0
230 DATA F0,00,00,00,00,00
```

*Listing 1. BASIC program.*

```
Loc.   Code          Label    Symbolic
Ø33A   D8            START    CLD
Ø33B   2Ø 36 E2               JSR CLSCR
Ø33E   A9 4D                  LDA #'M
Ø34Ø   2Ø A7 Ø3               JSR WRTW
Ø343   AØ ØØ                  LDY #ØØ
Ø345   A9 Ø4                  LDA #Ø4
Ø347   85 5C         INAD     STA ADRH
Ø349   2Ø D2 C9      DISP     JSR CRLF
Ø34C   A5 5C                  LDA ADRH
Ø34E   2Ø AD Ø3               JSR WRBY
Ø351   98                     TYA
Ø352   2Ø AD Ø3               JSR WRBY
Ø355   2Ø A5 Ø3               JSR SPTW
Ø358   B1 5B                  LDA (ADRL),Y
Ø35A   2Ø AD Ø3               JSR WRBY
Ø35D   2Ø E6 Ø3               JSR INPUT
Ø36Ø   EØ ØØ         CKINS    CPX #ØØ
Ø362   FØ 2F                  BEQ OPEN
Ø364   B5 1F                  LDA BUF-1,X
Ø366   C9 5E                  CMP #'↑
Ø368   DØ ØA                  BNE CKCOL
Ø36A   88                     DEY
Ø36B   CØ FF                  CPY #$FF
Ø36D   DØ DA                  BNE DISP
Ø36F   C6 5C                  DEC ADRH
Ø371   4C 49 Ø3               JMP DISP
Ø374   C9 3A         CKCOL    CMP #':
Ø376   DØ ØE                  BNE CKX
Ø378   CA                     DEX
Ø379   8A                     TXA
Ø37A   FØ BE                  BEQ START
Ø37C   2Ø C6 Ø3               JSR PKBY
Ø37F   A8                     TAY
Ø38Ø   2Ø C3 Ø3               JSR CKBY
Ø383   4C 47 Ø3               JMP INAD
Ø386   C9 58         CKX      CMP #'X
Ø388   FØ 5B                  BEQ EXIT
Ø38A   C9 53                  CMP #'S
Ø38C   FØ ØD                  BEQ SAVR
Ø38E   2Ø C6 Ø3               JSR PKBY
Ø391   91 5B                  STA (ADRL),Y
Ø393   C8            OPEN     INY
Ø394   DØ B3                  BNE DISP
Ø396   E6 5C                  INC ADRH
Ø398   4C 49 Ø3               JMP DISP
Ø39B   2Ø E6 Ø3      SAVR     JSR INPUT
Ø39E   86 EE         SAVIT    STX FNLEN
Ø3AØ   A2 ØØ                  LDX #ØØ
Ø3A2   4C B1 F6               JMP SAVE
Ø3A5                 ;        Subroutines follow
Ø3A5   A9 2Ø         SPTW     LDA #$2Ø
Ø3A7   2Ø D2 FF      WRTW     JSR WRT
Ø3AA   4C D2 FF      WRON     JMP WRT
Ø3AD                 ;
Ø3AD   48            WRBY     PHA
Ø3AE   4A                     LSR
Ø3AF   4A                     LSR
Ø3BØ   4A                     LSR
Ø3B1   4A                     LSR
Ø3B2   2Ø B8 Ø3               JSR WRASC
Ø3B5   68                     PLA
Ø3B6   29 ØF                  AND #$ØF
Ø3B8   18            WRASC    CLC
Ø3B9   C9 ØA                  CMP #$ØA
Ø3BB   9Ø Ø2                  BCC ASC
Ø3BD   69 Ø6                  ADC #Ø6
Ø3BF   69 3Ø         ASC      ADC #$3Ø
Ø3C1   DØ E7                  BNE WRON
Ø3C3                 ;
Ø3C3   8A            CKBY     TXA
Ø3C4   FØ 12                  BEQ BYRDY
Ø3C6   2Ø D9 Ø3      PKBY     JSR NIB
Ø3C9   EØ ØØ                  CPX #ØØ
Ø3CB   FØ ØB                  BEQ BYRDY
Ø3CD   85 FE                  STA LNIB
Ø3CF   2Ø D9 Ø3               JSR NIB
Ø3D2   ØA                     ASL
Ø3D3   ØA                     ASL
Ø3D4   ØA                     ASL
Ø3D5   ØA                     ASL
Ø3D6   Ø5 FE                  ORA LNIB
Ø3D8   6Ø            BYRDY    RTS
Ø3D9                 ;
Ø3D9   CA            NIB      DEX
Ø3DA   B5 2Ø                  LDA BUF,X
Ø3DC   18                     CLC
Ø3DD   C9 41                  CMP #$41
Ø3DF   9Ø Ø2                  BCC HEX
Ø3E1   69 Ø8                  ADC #Ø8
Ø3E3   29 ØF         HEX      AND #$ØF
Ø3E5   6Ø            EXIT     RTS
Ø3E6                 ;
Ø3E6   A2 ØØ         INPUT    LDX #ØØ
Ø3E8   2Ø A5 Ø3               JSR SPTW
Ø3EB   2Ø CF FF      RDON     JSR RDT
Ø3EE   C9 ØD                  CMP #$ØD
Ø3FØ   FØ F3                  BEQ EXIT
Ø3F2   EØ 1Ø                  CPX #$1Ø
Ø3F4   FØ F5                  BEQ RDON
Ø3F6   95 2Ø                  STA BUF,X
Ø3F8   E8                     INX
Ø3F9   DØ FØ                  BNE RDON
Ø3FB                 ;        Not used
Ø3FC                 ;        Not used
Ø3FD                 ;        Not used
Ø3FE                 ;        Not used
Ø3FF                 ;        Not used
```

Alphabetical list of label locations

| Label | Location | Label | Location |
|---|---|---|---|
| ADRH | ØØ5C | NIB | Ø3D9 |
| ADRL | ØØ5B | OPEN | Ø393 |
| ASC | Ø3BF | PKBY | Ø3C6 |
| BUF | ØØ2Ø | RDON | Ø3EB |
| BYRDY | Ø3D8 | RDT | FFCF |
| CKBY | Ø3C3 | SAVE | F6B1 |
| CKCOL | Ø374 | SAVIT | Ø39E |
| CKINS | Ø36Ø | SAVR | Ø39B |
| CKX | Ø386 | SPTW | Ø3A5 |
| CRLF | C9D2 | START | Ø33A |
| DISP | Ø349 | WRASC | Ø3B8 |
| EXIT | Ø3E5 | WRBY | Ø3AD |
| FNLEN | ØØEE | WRON | Ø3AA |
| HEX | Ø3E3 | WRT | FFD2 |
| INAD | Ø347 | WRTW | Ø3A7 |
| INPUT | Ø3E6 | | |
| LNIB | ØØFE | | |

Zero Page locations used

| Location | Label | Remarks |
|---|---|---|
| ØØ2Ø to ØØ2F | BUF | Input buffer for commands and File Name |
| ØØ5B | ADRL | Address Lo byte |
| ØØ5C | ADRH | Address Hi byte |
| ØØE5 | EADL | Ending address Lo byte for SAVE |
| ØØE6 | EADH | Ending address Hi byte for SAVE |
| ØØEE | FNLEN | File Name length for SAVE |
| ØØF1 | DN | Device Number for SAVE |
| ØØF7 | SADL | Starting address Lo byte for SAVE |
| ØØF8 | SADH | Starting address Hi byte for SAVE |
| ØØF9 | FNADL | File Name address Lo byte for SAVE |
| ØØFA | FNADH | File Name address Hi byte for SAVE |
| ØØFE | LNIB | Low nibble used in PKBY subroutine |

PET Operating System calls used

| Label | Location |
|---|---|
| CLSCR | E236 |
| CRLF | C9D2 |
| SAVE | F6B1 |
| RDT | FFCF |
| WRT | FFD2 |

*Listing 2. PET machine-language Mini Monitor program.*

the addresses as necessary. Once these modifications are made, type S and press the return key. The display should then look like this:

00FB XX S ()— Blinking cursor

You may now type the name of the program. Up to 16 characters are allowed for the program name, and quotes are not necessary. After you finish typing the name, press the return key. You should then see this message displayed:

PRESS PLAY & RECORD ON TAPE #1

When the save routine in finished, the tape will stop and the program will exit to the BASIC interpreter operating system to give you the familiar READY display. Programs saved on tape by Mini Monitor can be loaded or verified in the same way as programs recorded by BASIC. When saving programs with Mini Monitor, make sure that the starting address you enter is less than the ending address, because the program does not check this. Otherwise, it will be necessary to execute a power-off reset of the machine and your program will be lost. To save the Mini Monitor on tape, enter the values in Table 4 into the indicated addresses.

To avoid errors, use the following guide whenever you want to save a program using Mini Monitor:

| Address | |
|---|---|
| 00E5 | Enter ending address Lo byte |
| 00E6 | Enter ending address Hi byte |
| 00F1 | Enter 01 |
| 00F7 | Enter starting address Lo byte |
| 00F8 | Enter starting address Hi byte |
| 00F9 | Enter 20 |
| 00FA | Enter 00 |

Type S and press RETURN
Type program name and press RETURN

Once you have Mini Monitor on tape, you can load it whenever you need to. The program is loaded from tape by typing LOAD and pressing the return key, and then pressing the PLAY on tape #1. In a few seconds, you should see this display:

FOUND MINI MONITOR
LOADING

The program should only take a few seconds to load before the normal READY indication appears. Now type NEW and press the return key to reset the BASIC operating system pointers after the load operation. You may now enter BASIC programs as usual or use Mini Monitor.

If you already have a BASIC program in your machine that you would not want to lose by typing NEW, and want to load Mini Monitor, you must first type ?PEEK(124), PEEK(125) and press the return key. Write down the values and their respective locations, then load Mini Monitor in the normal manner. After the load is finished, type POKE 124,X:POKE 125,Y, where X and Y are the values you wrote down for their respective locations, and press the return key. Your program is now preserved, and Mini Monitor is ready for use.

Listing 2 is the machine-language program listing and includes an alphabetical list of label locations, zero-page locations used and the PET operating system calls used in the program. ■

*Merrill Lessley*
*4903 Northaven Avenue*
*San Diego, CA 92110*

# Easy-to-Build Computer-Controlled Triac Dimmer

*This hardware/software article presents this project in the right light.*

You can control the intensity of high-wattage lamps or the speed of universal ac motors with your computer by adopting the circuitry and software in this article. The dimmer works through conventional phase-control techniques and, because of a simple sensing circuit, even adjusts to variations in line voltage.

Design features include an optically isolated gating circuit that helps protect your favorite piece of hardware from destructive ground loops or other disasters. Power capacity is determined primarily by triac selection. Most triacs work well when attached to an acceptable heat sink and provided with adequate ventilation. With a 40 ampere triac and convection cooling, the unit comfortably drives a 200 watt load.

Besides controlling such high-current loads this dimmer is also a voltage-controlled device. This means that remote-control applications can be accomplished with minimal external wiring. Interfacing is accomplished through a digital-to-analog converter (DAC), which transmits a 0 to 5 volt dc signal to the dimmer's control card. Final ac output is a function of this signal, though not necessarily a linear one.

### Circuit Description

Before constructing the dimmer, you should understand how the circuit works. The schematic in Fig. 1 shows a small isolation transformer coupled to a full-wave bridge formed by diodes D1, D2, D3 and D4. The unfiltered output from this bridge acts as a gating signal for the zero-crossing detector created by Q1. Whenever the output from the bridge approaches a zero potential, Q1 produces a narrow and positive pulse, which is inverted by Q2. One-hundred and twenty such pulses occur every second.

This pulse chain continually resets the ramp signal generated by C3 charging itself through R5. Buffering this ramp is the source follower formed by operational amplifier (op amp) A in IC1. An oscilloscope connected to test point B reveals a linear ramp being reset each time the ac line signal crosses its zero point (Photo 1).

The ramp signal from op amp A travels to the non-inverting input (pin 10) of a voltage comparator (op amp B). This comparator

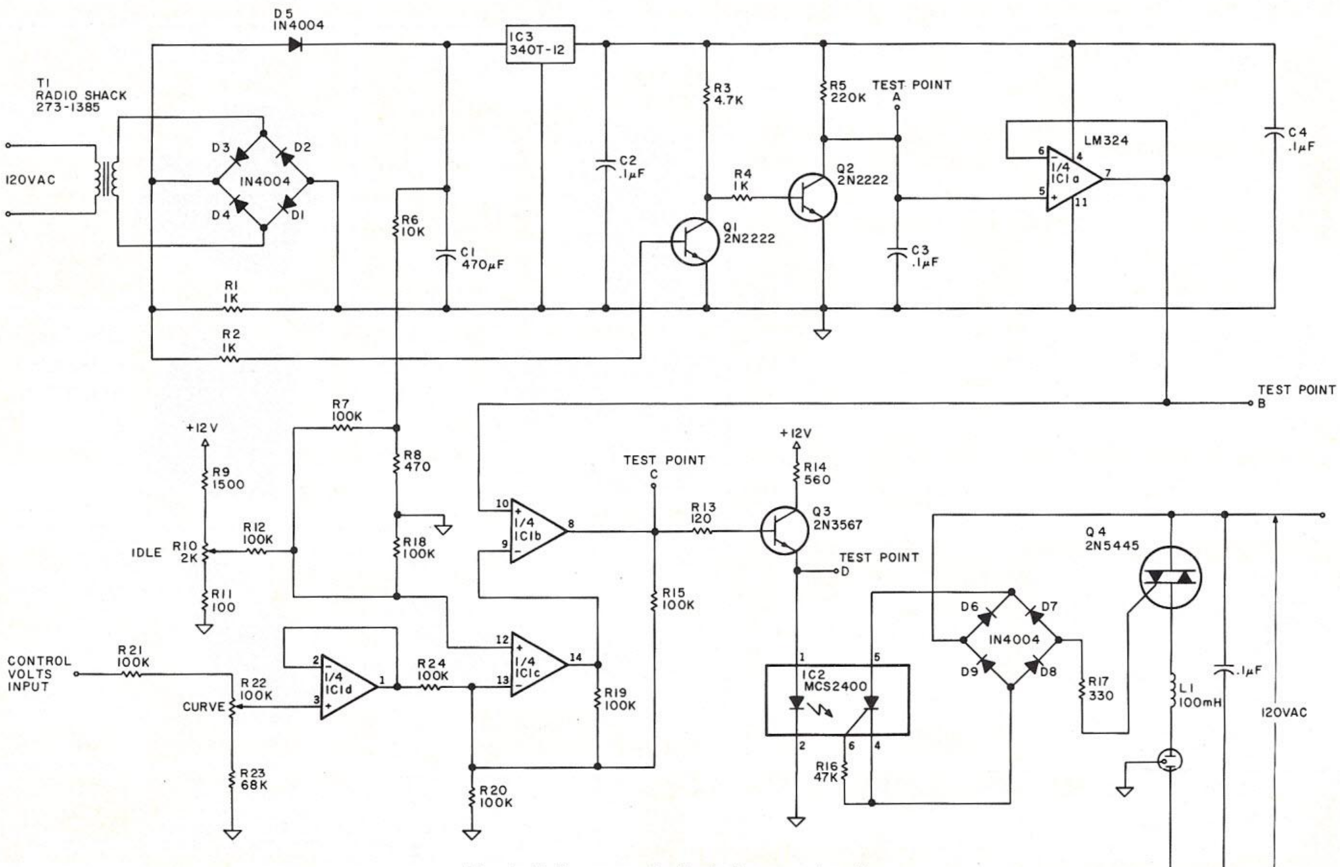



*Fig. 1. Voltage-controlled dimmer circuit.*

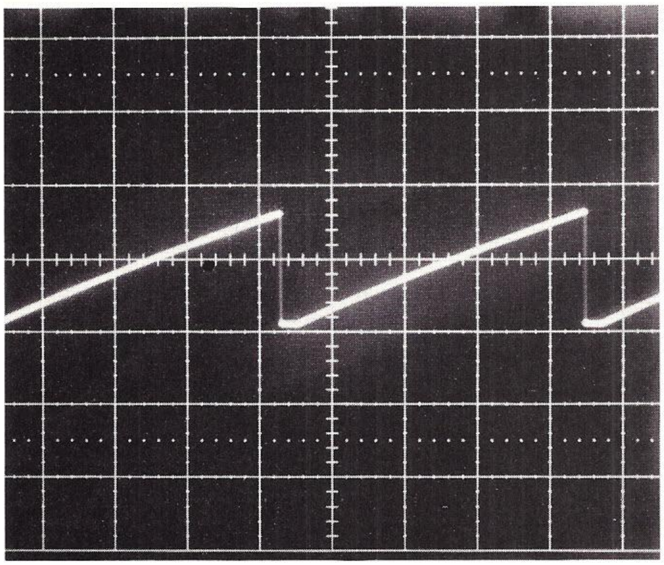


Photo 1. Time—2 ms/C. Volts—2 mV/C.

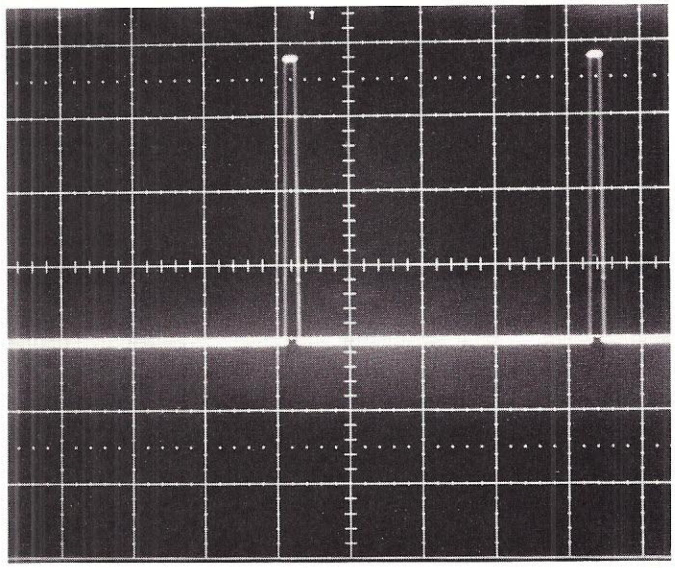


Photo 2. Time—2 ms/C. Volts—2 mV/C.

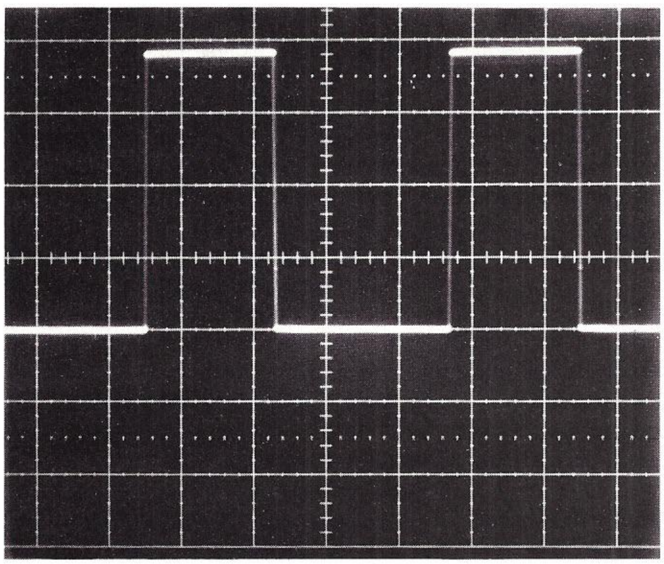


Photo 3. Time—2 ms/C. Volts—2 mV/C.

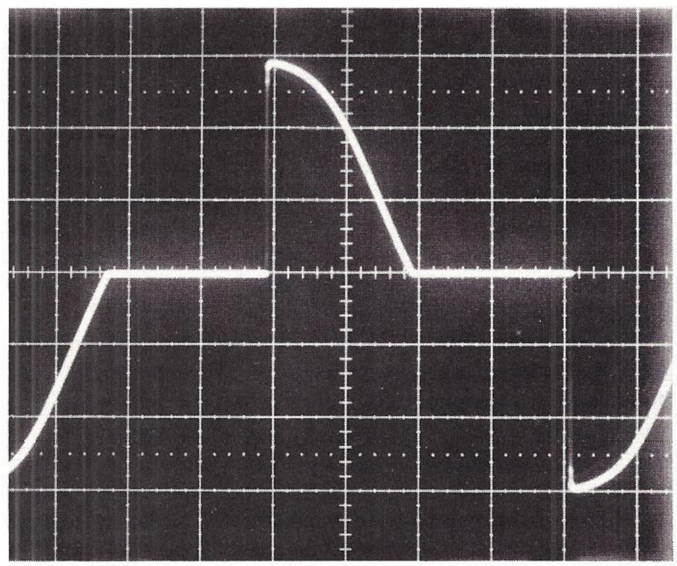


Photo 4. Time—2 ms/C. Volts—50 V/C.

swings into positive saturation as the ramp signal reaches a level almost equaling the reference voltage at pin 9. Resetting the ramp forces the comparator back into negative saturation. The width of the output pulse at pin 8, therefore, is determined by the voltage applied to pin 9. Q3 inverts this variable width pulse and correspondingly turns the LED in IC2 on and off.

To illustrate the importance of this variable width pulse, consider the following examples referenced to the positive half of the ac line signal. A voltage level at pin 9 slightly less than the ramp's maximum amplitude produces a narrow pulse at test point C (Photo 2). The output of this pulse is timed to occur as the positive swing of the sine wave approaches its zero potential. Lowering the voltage at pin 9 produces a wider pulse at the comparator's output (Photo 3), whose leading edge occurs earlier in the sine wave's positive swing. The sine wave's negative half duplicates this pulse pattern.

Pulses from the comparator turn the LED on and off and, therefore, trigger the light-activated silicon-controlled rectifier (LASCR) in IC2. This LASCR, working in conjunction with a full-wave bridge (diodes D6, D7, D8 and D9), helps to generate a series of pulses that eventually reaches the triac's gate through R18. And this finally switches the triac on and off so that current flows through the load.

In summary, the amount of current passing through the load relates directly to the triac's rapid on-off action. A low-level signal at pin 9 triggers the triac early in the sine wave when more current can ultimately pass through the load. Conversely, increasing the dc potential at pin 9 reduces the triac's on time and decreases its output capability.

Photo 4 illustrates how a triac's switching action literally slices up the ac sine wave. The pulse chain producing this example is sampled in Photo 2. In this case, the triac is driving a 300 watt incandescent load at 80 volts true rms.

At this point, the circuit represents a simple voltage-controlled dimmer. Unfortunately, an inverse relationship exists between control voltage and actual dimmer output. Solving this inconvenience requires the addition of op amp C. Output from this amplifier equals the voltage at pin 12 minus the voltage at pin 13.

The circuit reveals that the minimal signal at pin 12 must normally be slightly less than the ramp's maximum amplitude. R10 provides this signal and correspondingly acts as the dimmer's idle or low-end adjustment: The dimmer will start to idle if 0 volts appear at pin 13 and R10 supplies pin 12 with a dc signal just below the ramp's maximum amplitude. Raising the voltage at

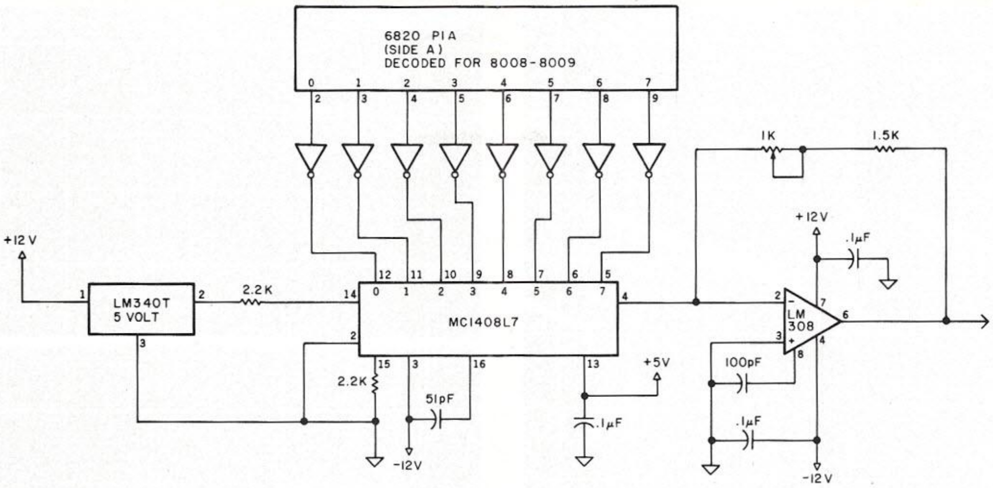



*Fig. 2. Digital-to-analog converter circuit.*

pin 13 increases the dimmer's output. Op amp D, configured as a source follower, buffers the dc input signal at pin 3, IC1.

By adding op amp C, the dimmer's output becomes a positive function of control voltage input. At 0 control volts, the dimmer idles or remains off; at 5 control volts, the dimmer reaches full power. Op amp C further serves as a sensing circuit along with R6, R7 and R8, all of which monitor line-voltage variations.

An appreciable drop in line voltage decreases the dc potential at pin 12 via R6 and R8. And, because of the summing mode of op amp C, the voltage at pin 9 is also reduced. It is this reduction that forces the triac to advance its firing angle and thereby provide compensation for apparent line loss. Incidentally, R8 sets the sensitivity of this circuit.

This means that computer control of the dimmer's output can be achieved by connecting a digital-to-analog converter to the dimmer's control-voltage input. Since this arrangement only requires two wires, you can easily install the dimmer in remote locations. Almost any DAC will do, providing it outputs a 0-to-5-volt signal. The converter in Fig. 2 works well for general applications and is driven by a simple PIA port. Note the PIA's inverted output. On my system, this inversion helps because a manual reset drives the PIA outputs high. Inverting the port adjusts the DAC's output to 0 volts after each reset.

### Construction

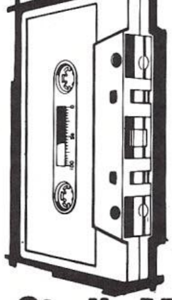
Constructing the dimmer is simple. You can print a PC board from the actual-size foil pattern in Fig. 3; Fig. 4 details the correct parts layout. An assembled board is pictured in Photo 5, and Fig. 5 presents the final wiring diagram.

Avoid connecting the dimmer's power supply common to the ac line ground. Also remember to polarize the plug by connecting the black wire to the hot side of the ac line and the white wire to the neutral side. The all-important safety ground, the green wire, must be attached to the dimmer's metal enclosure. Follow this procedure to protect both you and your equipment from dangerous electrical shocks.

Take special care to locate the dimmer's power outlet on the neutral side of the triac to provide a specific safety measure intended to reduce the possibility of inadvertent shock. Wiring the plug into the triac's hot side increases the danger of electrical shock because the full ac line power is always just a fingertip away, even with an inactive or idling triac. Placing the plug on the neutral side forces both sides of the outlet to remain essentially at ground potential whenever the triac is off or in a high-impedance state.

The triac's rapid on-off switching produces current surges that generate significant radio-frequency interference (rfi). Long





circuit wires act as transmitting antennas for such noise. To counteract this, shield or twist the triac's gate leads and keep them short. Rfi can also travel down the ac line. This can be largely corrected by utilizing a 100 mH choke (L1) and a bypass capacitor (C5).

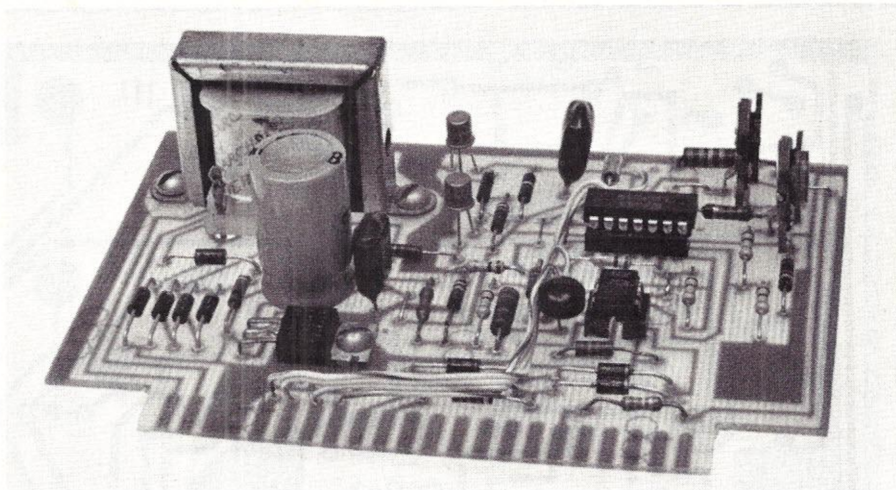


*Photo 5. An assembled circuit board.*

### Testing and Adjustment

Test and adjust the dimmer before you connect it to your computer. Start testing and adjusting procedures by centering pots R10 and R22. Connect an ordinary 60 watt lamp to the dimmer's output. Plug the dimmer in and adjust R10 until the lamp just barely begins to glow. Unplug the dimmer and replace the lamp with a load approximately equal to 50 percent of the dimmer's intended capacity. Apply power and readjust R10.

If you have a spare 5 volt power supply, the next step will be easy. Just attach a 10k potentiometer across the supply leads. Adjust the pot for a 2.5 volt output. Connect this signal to the dimmer's control voltage input and adjust R22 for 79 true root mean square (rms) volts across the load. Making this adjustment requires a true rms meter; ordinary ac meters are designed to measure relatively smooth sine waves and not the fragmented signals produced by triacs.

Producing the 79 volt true rms value at a 2.5 volt control signal helps produce a satisfactory visual curve for the dimmer. Because the human eye perceives light in a somewhat logarithmic fashion, the smooth control of incandescent loads requires that a nonlinear relationship exist between control voltage and true rms output.

Put another way, calibrating in this manner forces the dimmer's luminary output to appear as if it were actually a linear function of control voltage input. A more comprehensive explanation of this phenomenon is beyond the scope of this article. You should also realize that this adjustment slightly limits the dimmer's highest output to about 115 volts, an insignificant loss with incandescent loads.

To calibrate without a true rms meter, increase the control voltage to 5 volts and attach a standard ac meter across the load. Turn R22 fully clockwise. The meter should



✓ Reader Service index—page 241

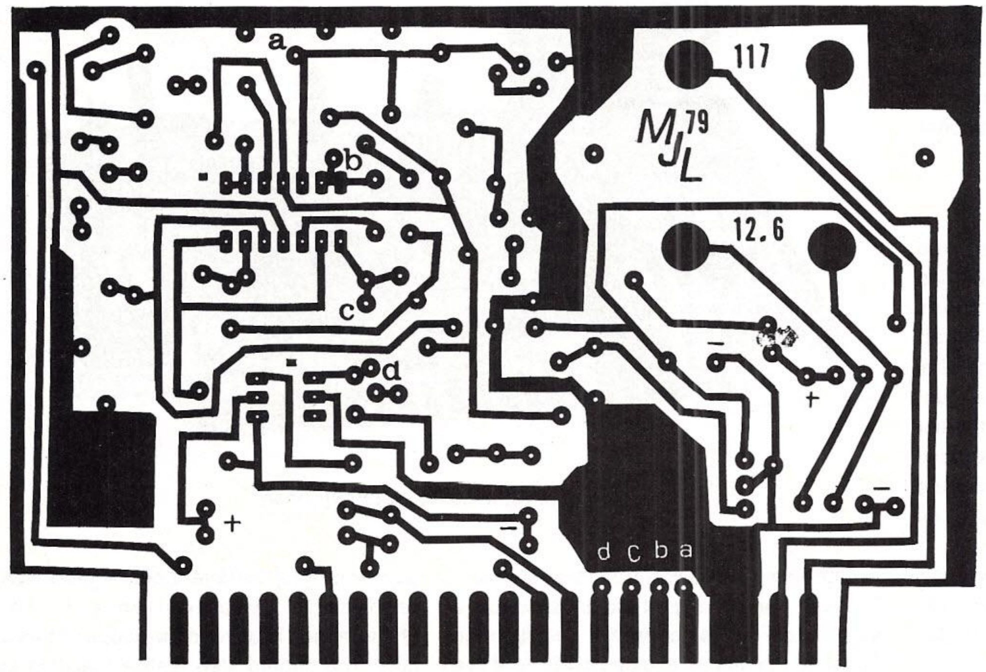



*Fig. 3. Full size PC board foil pattern.*

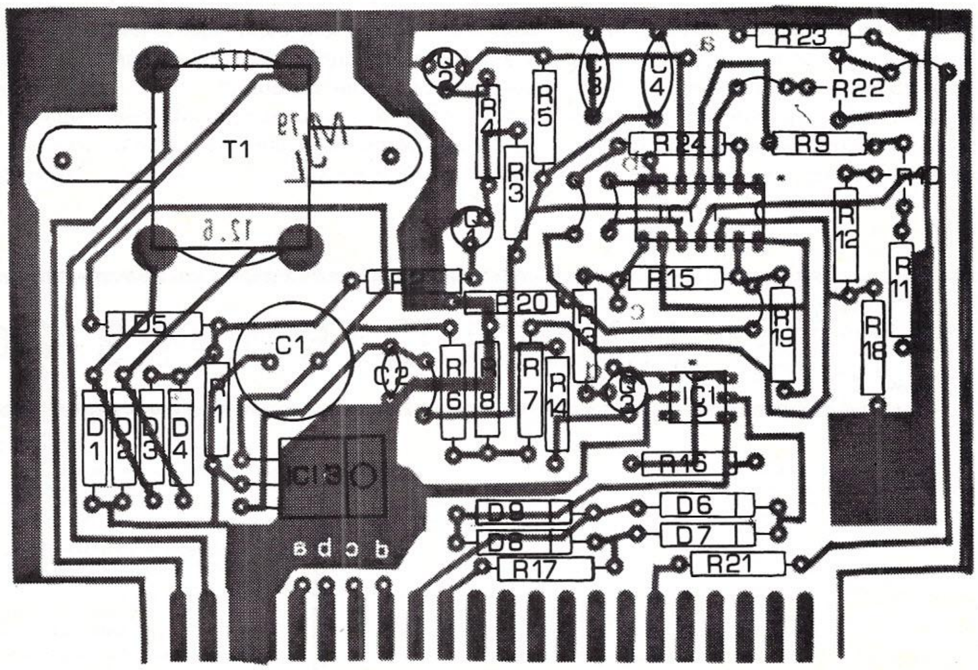



*Fig. 4.*

read between 115 and 120 volts. Now slowly rotate R22 counterclockwise until the meter indicates a 2 volt decrease from the previous reading. A fairly accurate curve can be achieved through this alternate procedure.

Alignment for universal ac motors is largely a matter of personal preference. Just remember that R10 adjusts the dimmer's idle level, and R22 determines its maximum output. These controls can also establish a confined operating range for the dimmer. R10, for example, might be set so that the dimmer's output will never fall below half power. Computer control would then range from half to full power. Also, when working with an electric motor, avoid overheating conditions that occur when a motor stalls or operates too slowly.

After testing and alignment, check again for wiring errors that might produce ground loops. Use an ohmmeter to make certain that no conductive paths exist between the ac line plug pins and the control voltage wires. If the dimmer passes this test, it can be attached to the computer's DAC.

Fig. 6 illustrates a convenient way of making this connection. Flexibility is gained through S1, which affords you the opportunity to select manual, computer or external control. Making this connection prepares the dimmer for actual software implementation.

**Software**

My system is an odd blend of parts, but, in general, it likes to assume the identity of a 6800-based unit with a MIKBUG monitor. And, as I've said, the computer's DAC is driven by a PIA at $8008 and $8009.

Systems configured like this can use Listing 1 to program a 6820 PIA to act as an eight-bit latching port. Load the program and place the hexadecimal data to be transmitted into memory location $000D. After this, load the starting address ($0000) into $A048 and $A049. Type G, and the port will respond by sending the DAC its data.

Fig. 5. Final wiring diagram.

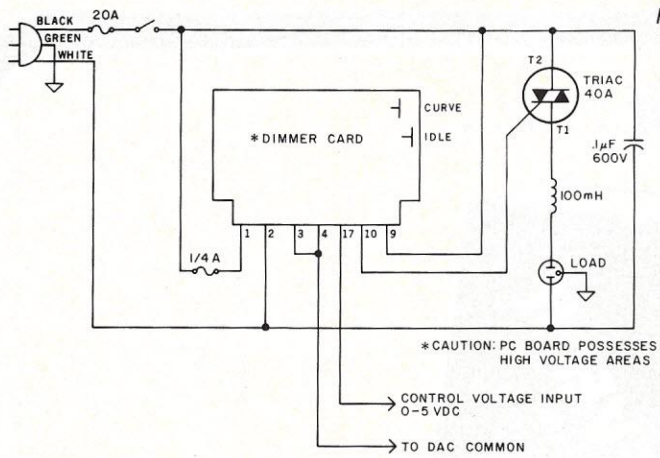
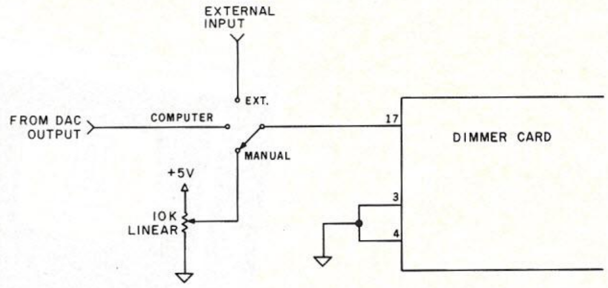



Fig. 6. Interface selector circuit.




```
0000  7F  8009   CLR    $8009          Initialize PIA into Latching
0003  86  FF     LDA A  #$FF           Port
0005  B7  8008   STA A  $8008
0008  86  04     LDA A  #$04
000A  B7  8009   STA A  $8009
000D  86  XX     LDA A  #$XX (DATA)    Load and Send Data
000F  B7  8008   STA A  $8008
0012  3F         SWI                   (Change to 39 for User Routine)
```

Listing 1. PIA routine.

```
 10  REM DIMMER CONTROL ROUTINE
 20  REM ESTABLISH USER PROGRAM LOCATION
 30  POKE (103,30)
 40  POKE (104,175)
 50  REM NOW INPUT DIMMER LEVEL
 60  INPUT "LEVEL",X
 70  IF X>255 THEN 60
 80  REM INVERT THIS VALUE
 90  LET Z=255-X
100  POKE (7869,Z)
110  REM CALL USER ROUTINE
120  LET A=USER(Z)
130  REM PRINT DIMMER LEVEL ON SCREEN
140  LET Q=255-A
150  PRINT
160  PRINT Q
170  GOTO 60
180  END
```

Listing 2.

Relocating the routine is no problem, but remember to clearly identify the data byte's storage location. For PIAs not at $8008, alter the program's address calls. Notice, too, that the circuit in Fig. 2 inverts the PIA's output, which means that data byte must be a complement value.

If you change the last instruction of the listing to RTI (39), the routine will work nicely in the user space of SWTP BASIC. To program dimmer output, poke an appropriate decimal value (0 to 255) into the data location and jump to the user routine. Control returns to the BASIC interpreter after the PIA has latched. Dimmer output remains stable until another command reactivates the DAC.

Listing 2 is an example of how POKE commands can interact with Listing 1. In this case, the program has been relocated to $1EAF, the beginning of user space in SWTP BASIC (ver 2.0). The first two instructions poke the address of the user routine into memory locations $67 and $68. This tells the interpreter where to transfer control when using the user command. The input instruction requests a dimmer "level," which must be entered in decimal format (0 to 255). This value is inverted and poked into $1EBD. The dimmer level then appears on the terminal's screen, and the program requests another input.

User routines can be frustrating, especially when coupled with POKE instructions. One mistake can cause the interpreter to self-destruct or damage the user program. Some protection can be afforded by carefully defining user space in advance. With SWTP BASIC, for example, start the process by initializing the interpreter to





✓240

verify the cassette load. After this, manually reset the computer and allocate user space by placing the desired byte in $104E. Complete the sequence by exercising a hard rather than a soft start. This procedure forces BASIC to recognize the new parameters that distinguish user space from memory dedicated to normal variable and program storage.

The principles expressed in these examples can be applied to more complex programs that simultaneously control several dimmers. To do so, however, requires that the detailed work of controlling and selecting dimmers must be accomplished by efficient user routines. The BASIC interpreter may be easy to work with, but, in general, it is too slow to directly implement complex and fast-shifting lighting effects.

### Conclusion

Here are some brief comments concerning the design philosophy surrounding this dimmer. One approach was the creation of a purely digital dimmer with the ability to interpret serial or parallel data commands; for certain applications this seemed keenly advantageous. In my case, however, a mixture of digital and linear techniques worked best. The need to utilize long remote-control lines was the deciding factor. To run 16 analog lines from a computer is infinitely easier than extending and decoding 16 serial lines.

Also, consider the problems associated with implementing manual override when using serial data. At least the linear circuit makes manual or independent control easy to work with. You can even intermix manual and computer control by presetting the independent potentiometer and driving it with the computer's DAC. Try such intermixing with an ACIA or UART.

In retrospect, then, this article shows how digital and linear techniques can merge to produce a versatile tool with many applications.■

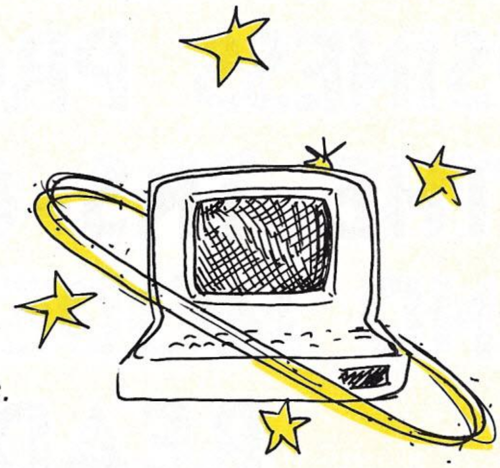

The author wishes to thank the San Diego State University Foundation, which helped support this work through a faculty grant-in-aid.

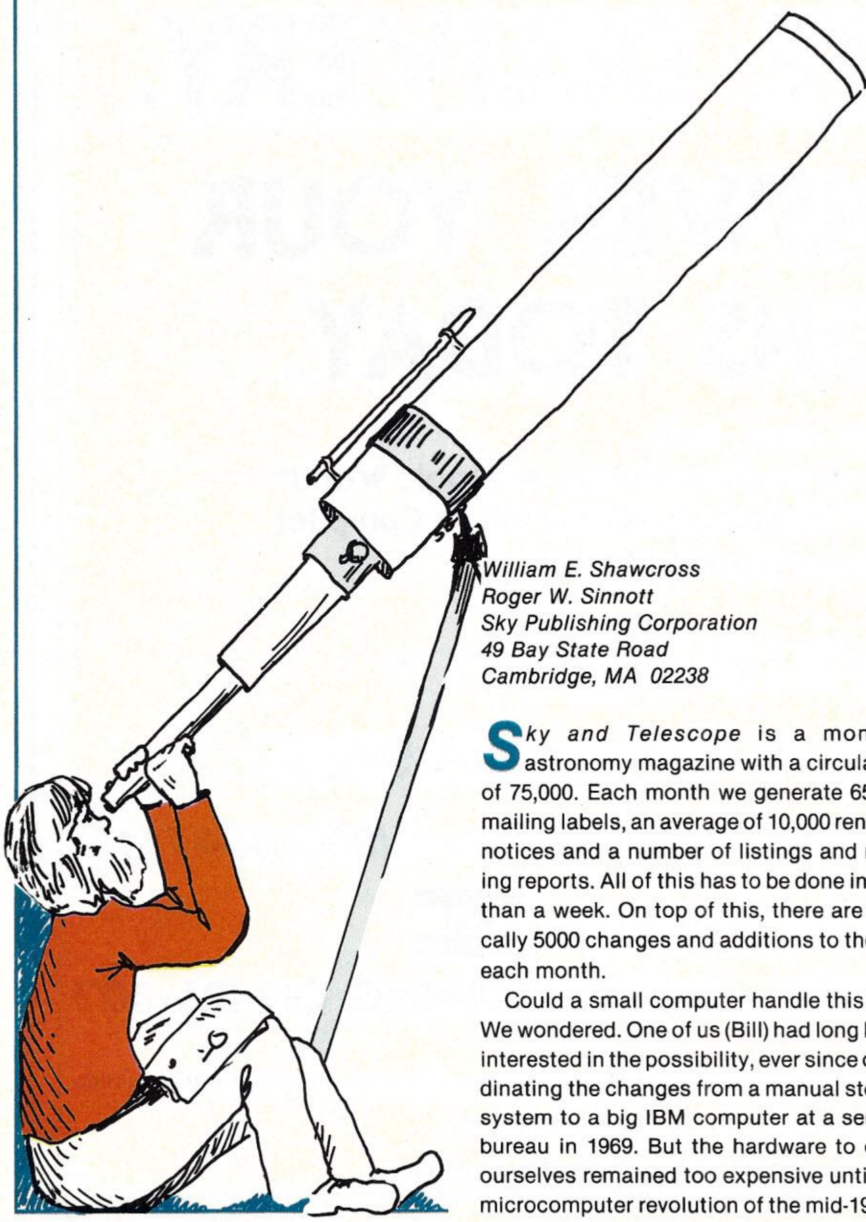

| | |
|---|---|
| D1 through D9—1N4004 diode | R15—100,000 Ohms |
| C1—470uF 50V electrolytic capacitor | R16—47,000 Ohms |
| C2, C3, C4—.1uF 50V ceramic | R17—330 Ohms |
| C5—.1uF 600V | R18 through R21—100,000 Ohms |
| R1—1000 Ohms | R22—100,000 Ohms PC trimmer |
| R2—1000 Ohms | R23—68,000 Ohms |
| R3—4700 Ohms | Q1, Q2—2N2222 npn transistor |
| R4—1000 Ohms | Q3—2N3567 npn transistor |
| R5—220,000 Ohms | Q4—2N5445 40A triac, or T6420D (isolated stud 40A triac) |
| R6—10,000 Ohms 1% | L1—100uH choke (current rating dependent upon triac selection) |
| R7—100,000 Ohms | IC1—LM324 quad op amp |
| R8—470 Ohms | IC2—LM340T-12, +12V regulator |
| R9—1500 Ohms | IC3—MCS2400 optical coupler |
| R10—2000 Ohms PC trimmer | T1—power transformer, 12.6VAC 300mA secondary, PC mount (Radio Shack 273-1385) |
| R11—100 Ohms | |
| R12—100,000 Ohms | |
| R13—120 Ohms | |
| R14—560 Ohms | |

*Parts list. All resistors ¼ watt 10 percent, unless otherwise noted.*

# OSI In the Sky

*A heavenly approach to handle subscriptions.*





*William E. Shawcross*
*Roger W. Sinnott*
*Sky Publishing Corporation*
*49 Bay State Road*
*Cambridge, MA 02238*

*Sky and Telescope* is a monthly astronomy magazine with a circulation of 75,000. Each month we generate 65,000 mailing labels, an average of 10,000 renewal notices and a number of listings and mailing reports. All of this has to be done in less than a week. On top of this, there are typically 5000 changes and additions to the file each month.

Could a small computer handle this job? We wondered. One of us (Bill) had long been interested in the possibility, ever since coordinating the changes from a manual stencil system to a big IBM computer at a service bureau in 1969. But the hardware to do it ourselves remained too expensive until the microcomputer revolution of the mid-1970s.

**Out of the Blue**

Roughly three years ago, Ohio Scientific introduced the C3-B, a rack-mounted microcomputer with dual eight-inch floppy-disk drives and a 74-megabyte Winchester hard disk. Our data base, then almost ten megabytes, had been too large for floppies, but the Winchester was just right. We were fortunate to have an OSI dealer in Cambridge, and so we visited Robert Rivers at the Computer Shop. A demonstration convinced us that the machine had enough speed and capability for our needs.

And so a year ago, after 1 1/2 years of preparation, we transferred our mailing list from the IBM 360/30 at the service bureau to our own in-house OSI computer. We were looking primarily for convenience; we didn't want to rely on the mails to get our subscription material (punched paper tape) back and forth from the service bureau. With our own computer, we would be able to put new subscribers on the system the day their names came in. We could also let the computer do a number of office jobs that for years we had done by hand.

The major unknowns were reliability and file backup, which are critical in converting to an in-house computer. At the time, cartridge-tape backup units were not available, and standard tape drives with formatters cost as much as the C3-B.

A magazine article on computer repair led us to the solution of both reliability and backup—get two identical computers. This would let us keep the files in separate places (in our case, in separate buildings). Regular copying from one machine to the other would keep the backup file within a month of being up-to-date. And circuit boards could be swapped easily in a pinch.

On this basis we took the plunge. We

*The new OSI C3-B computer, Printronix 300 and a Soroc terminal by which the same job is now done more efficiently in-house at* Sky and Telescope. *(Photos by Dennis diCicco)*

bought the first computer in August 1978, and the second one the following February. We ordered a Printronix 600 line printer (600 lines per minute) but eventually had to settle for a Printronix 300. We bought a used Centronics 100 printer with the first computer so we could do development work. Our suite of equipment was rounded out with a Tab miniburster (with slitter), Tab decollator, a Hazeltine 1500 terminal and three Soroc IQ-120 terminals. The total cost was around $45,000.

Getting the equipment was just the start. The real effort had to go into programming, which proved very time-consuming.

Neither of us was completely new to computers. Bill had gone to programming school and had put together one of the very first Altair 8800 kits. Roger had bought the Altair from Bill and learned North Star BASIC on it. Bill took over project management and definition, and Roger gave up some of his editorial duties so he could spend about half his time on programming. The project took a whole year.

In the late 1960s, Bill had helped define the functions of the mailing package used by our service bureau, and therefore knew what it did and how it did it. We assumed it would be a fairly easy matter to duplicate the functions of the big computer. But there were two areas in which we badly underestimated how much work we had to do.

**Software Problem and Solution**

The first of these was learning, in minute detail, how the Microsoft BASIC interpreter interacts with large files on the OSI 74-megabyte disk. For most hacking this is not a serious matter, but when you're trying to design as efficient a system as possible, everything counts. For example, Roger eventually found a way of copying large files that runs three times faster than the copying utility supplied by OSI.

On a large mainframe computer, a magazine subscription list is usually kept in serial order on a reel of magnetic tape. Changes are input, sorted and then processed against the tape. A new tape is made (on another tape drive) incorporating the changes. Such sorting and merging, however, is utterly impractical on a micro with a BASIC interpreter. We guessed that to merge two files to a third place on the disk would take us at least 20 hours for each weekly update.

A double-size file that stays put on the disk was our solution. Here, on the average, every other record is a blank, or "hole." Since our mailing list is in zip-code order, and names are alphabetical within each zip code, a binary search of about 17 seeks will find a record that needs revising in about six seconds. If a new subscriber is to be inserted, the same amount of time will locate the appropriate hole. Occasionally, a bit of spreading is needed to make room for the new person, if some records are bunched together. Our updating program does this automatically.

In essence, we have an on-line system that is run from floppy disks rather than directly from the keyboard. Since an audit-trail report is needed for all file transactions to satisfy the Internal Revenue Service and the postal system, putting the work on a floppy first means that the line printer does not have to be running except when the floppies are being processed against the master file (about 30 minutes, at the end of each day's work). And since the data base is copied to the backup computer only once a month, we save four weeks of floppies in case the primary computer should ever fail. A floppy-based system thus has many im-

*The service-bureau computer room, with numerous tape drives, where the* Sky and Telescope *subscriptions were processed between 1969 and 1979. It is operated by Computac, Inc., in West Lebanon, NH. The* Sky and Telescope *circulation department prepared punched paper tape (on a Friden Flexowriter) that was delivered to Computac weekly, containing all the names of new subscribers or renewals in machine-readable form.*

portant advantages.

### File Transfer

Our second major problem was getting our data base out of IBM-land and onto our own machine. We thought of a number of possibilities: punched paper tape (16 miles of it), the telephone (90 hours of long-distance charges), a 1/2-inch computer tape (rental drives not readily interfaced to OSI) and rekeyboarding all the names and addresses (1000 hours for a good typist).

Floppy disks seemed the only possibility, but there was a catch here too. Our service bureau could supply the file on eight-inch floppies, recorded in the standard IBM format of 128 bytes per sector, 26 sectors per track. OSI uses the same hardware but a different format. In fact, OSI's format seems to be unique in the industry, so buying software from other sources is difficult.

Here Roger's Altair came to the rescue. We bought a Tarbell S-100 disk-controller board and attached the OSI floppy drives to the Altair. A word of caution: OSI supplies the −5 volts from the CPU power supply through the connecting ribbon cable, rather than from the floppies' own power supply. This caused us a major headache until Roger traced the problem down.

An assembly-language routine for the Altair read each track-size block of 13 names and addresses from a floppy into RAM, inserted ASCII carriage returns in all the right places and sent them out a serial port at 9600 baud to the OSI computer. There, a BASIC program accepted each character string in turn and sent a handshaking character back to the Altair when it was ready for the next string. (This was necessary because the Microsoft BASIC interpreter seemed to take unpredictable amounts of time to assimilate the strings as they came in.)

When 13 names and addresses had been transferred, the OSI computer paused for half a second to store them sequentially on the hard disk and asked for more. The Altair then read in the next higher track, and so on. Proceeding in this manner, the two computers worked without intervention for about seven minutes to transfer all 949 names on a floppy. It was a complicated operation, to be sure, but when things were going smoothly it was spectacular to watch!

In this way the file was transferred, at a mere cost of $250 for the Tarbell board and $200 to the service bureau to have them put the file on 69 floppy disks we had provided (and which we are now reusing for our daily work). We did the transfer when our other software was essentially done, so that we could try out our own system in parallel with that of the service bureau for a couple of months to be sure the results were the same. Six months is usually recommended, but the duplication of several thousand transactions a month was putting a severe strain on our circulation department of just three people. Fortunately, all went well, and we were able to complete the changeover in November 1979. We have been self-sufficient ever since.

### System Operation

As it now stands, our system lets us enter on floppies new subscriptions, renewals, address changes and deletions. This can be done with more than one terminal at once, if necessary, thanks to the OSI time-sharing option. The operator can request adhesive mailing labels to be printed so the current issue of the magazine can be sent immediately—something that used to be done by hand.

We also have two programs for locating particular people in the file. The first uses the same binary search technique mentioned earlier. If you know a subscriber's zip code and a few letters of the last name, the record can be called up in six to seven seconds. (Sometimes the program tells you that the name is not currently in the file.)

The second method sweeps through the file from any starting place to the end, looking for a specified character string of up to 32 characters. Thus, for example, if Ebenezer Jones lives somewhere in California, it makes sense to use his first name as the search string and to start looking at zip code 90000. All too often, a person sends in a change of address with the old address missing! This kind of search, from one end of our 23 megabyte file to the other, can take as long as 18 minutes.

Along with renewal notices and mailing labels, the system produces a monthly geographical breakdown by zip code, used for postal reports, and various file dumps. We also make a printout each month of the names of people who decided not to renew; this list can be used at a later date, in case we want to do a selective mail campaign.

We still need to speed up the file transfer between the two machines. This procedure now takes about 3 1/2 hours at 19,200 baud over a 100-foot coaxial cable. The OSI serial ports can be jumpered for 250,000 or even 500,000 baud as soon as we find time to write the assembly-language handlers for synchronous data transfer. This should cut the time to under half an hour for the 23 megabytes.

### Extraterrestrial Applications

We have found other uses for all this computing equipment. One nice feature of the Printronix 300 is its ability to serve as a plotter. Roger has written a program that prints out a chart showing the motions of Saturn's moons; all we do is add labels and send it off to the camera department for use in *Sky and Telescope*. Most days of the month, the second (backup) computer is available for use by our scientific staff.

Another area in which the computer helps is our advertising billing and accounts receivable—a sizable portion of our income. For this we bought the OSI Data Management System nucleus and modified parts of it to print our bill forms. This software may not be the fastest or most elegant in the world, but it is dependable, easy to work with and reasonably priced ($300). It has general-report writers and file handlers, allowing easy maintenance of a small advertiser mailing list. We also have a separate dealers file for addressing the new price lists of the books we sell.

### A Successful Flight

We have faced few difficulties since the changeover—and none that has caused us to miss a schedule. This is probably due in large measure to the fact that we know a lot about our equipment, and everything there is to know about our subscription programs. The usual memory glitches (chips go bad from time to time) and a dead LED in a floppy drive (it couldn't find the index hole) are the only equipment failures we have had in two years.

We had some strange error messages and the like when the computer-room temperature rose above 80 degrees Fahrenheit. The equipment behaves strangely at such temperatures, so we now use air conditioning when needed to keep the room from getting warmer than 75 degrees.

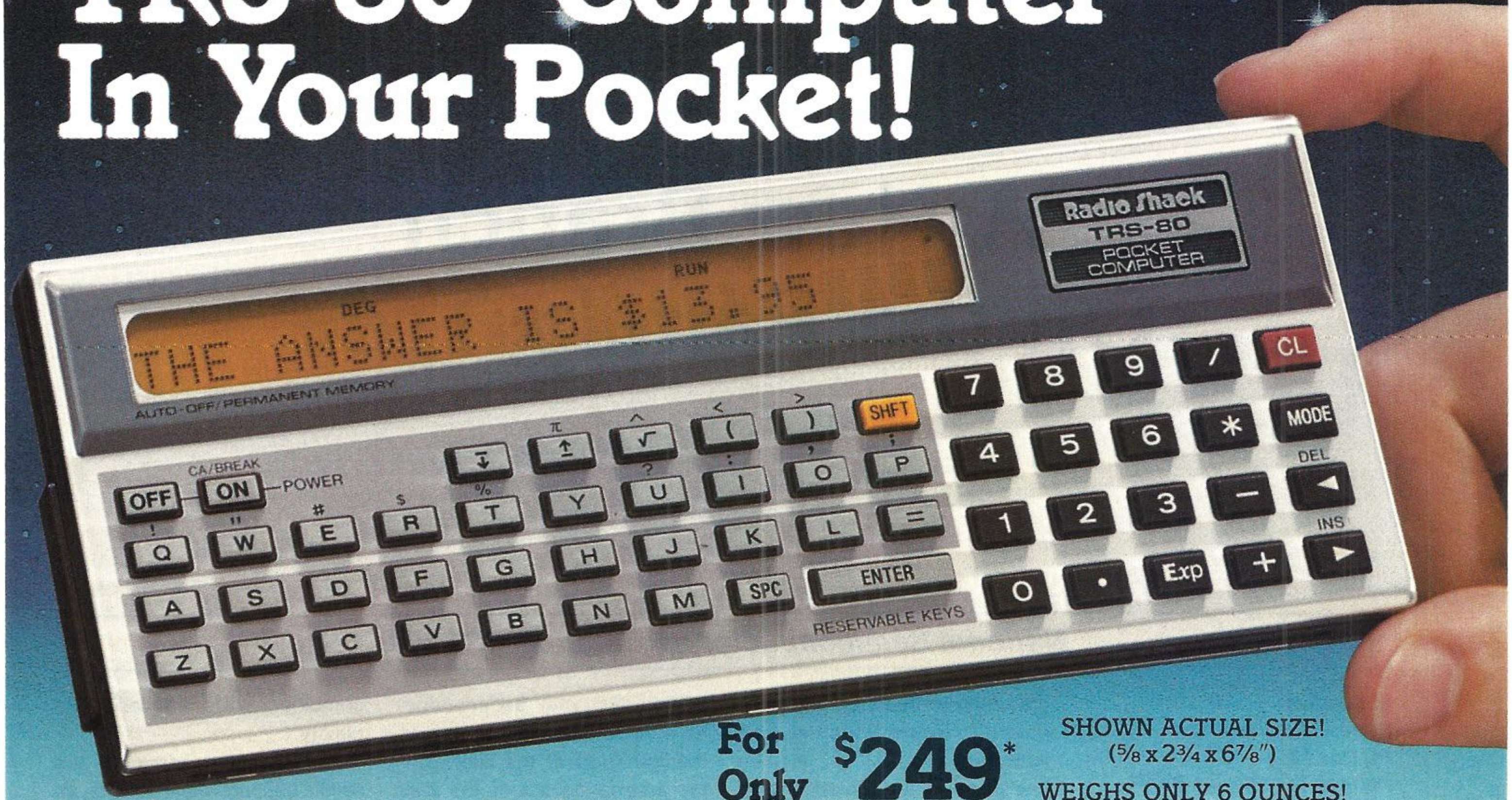
Also, though the OSI manuals warn against shutting off the CPU before the hard disk, we have inevitably done this accidentally a couple of times. Sometimes it doesn't have any effect, and other times it wipes out the operating system on the disk and forces us to get the backup floppy out of a vault. (Why can't the equipment be designed so that it doesn't matter how you shut it off?)

A dependable dealer nearby is a great help. Bob Rivers at the Cambridge Computer Shop and Ed Craddock at the Boston branch have helped us often both with suggestions and new software releases.

Make a careful estimate of how long it will take to program an application and then double or triple the time. Don't get yourself into a corner with rigid scheduling—equipment is seldom delivered as soon as you expect, and software is almost never released on the date first announced. Anticipate being discouraged and ready to give up every now and then, particularly during the development phase.

And, above all, enjoy the satisfaction of a difficult project finally brought to fruition. ■

# Now You Can Put a TRS-80® Computer In Your Pocket!




For Only $249*

SHOWN ACTUAL SIZE! (5/8 x 2¾ x 6⅞")

WEIGHS ONLY 6 OUNCES!

## Another Radio Shack Personal Computer Breakthrough!

Now you can carry computer power wherever you go. And with the TRS-80 Pocket Computer, you can create your own programs or purchase ours. These are available now: Civil Engineering, Aviation, Business Statistics, Games, Personal Finance, Math Drill, and Real Estate.

***Look At These Exciting Features!*** 57-key alphanumeric keyboard. Big LCD display that scrolls left and right for program line entry and editing, and steps up and down for program listing. Programmable in an enhanced, easy-to-learn BASIC language. Built-in arithmetic functions including trig and inverse trig (with readout in degrees, radians or gradians) log, exponent, square root, angular conversions, integers and absolute values. Accuracy is to 10 digits and it can handle 2-digit exponents. With array and 7-character string-handling ability you've really got a handful of computational power!

***Programs and Data Retained in Memory When Power Is Off!*** The 1424-step memory is automatically partitioned for program and data storage. There's a 26-data element memory and 48-step reservable memory.

***Available Now!*** Exclusively at Radio Shack stores, dealers and Computer Centers. Complete with carry case, manual, batteries. Hurry, because everyone will want this sensational new computer. And what could make a better or more unique Christmas gift?

### Nothing Else You Can Put In Your Pocket Can Do All This!

```
10 Print "SHELL METZNER SORT": FOR X = 1 TO 100:
   PAUSE "DATA ITEM #";X:INPUT A(X + 100): IF A(X + 100
   )<0 GOTO 25
20 NEXT X
25 M = X - 1
30 M = INT(M/2): IF M = 0 GOTO 107
40 J = 1: K = X - M - 1
50 I = J
60 BEEP 1: L = I + M: IF (A(I + 100)< = A(L + 100))
   GOTO100
70 T = A(I + 100): A(I + 100) = A(L + 100):
   A(L + 100) = T: I = I - M: IF I<1 GOTO 100
90 GOTO 60
100 J = J + 1: IF J>K GOTO 30
105 GOTO 50
107 BEEP 5: INPUT "PRESS ENTER FOR LIST"; A
110 FOR I = 1TOX - 1: J = I + 100: PAUSE "DATA ITEM #";
    USING "####"; I; " "; A (J): NEXT I
```

**Send Me Your FREE TRS-80 Computer Catalog!**

Radio Shack, Dept. 81-A-23
1300 One Tandy Center
Fort Worth, Texas 76102

NAME

TITLE

FIRM

STREET

CITY STATE

ZIP PHONE

*Retail prices may vary from store to store. Optional Cassette Interface, $49. Minisette-9 Cassette Recorder, $79.95.

**Radio Shack®**

**The biggest name in little computers™**

# A New Branch On the Family Tree

*National Semiconductor sprouts the NSC800.*


*Ken Barbier*
*Borrego Engineering*
*PO Box 1253*
*Borrego Springs, CA 92004*


The recently announced NSC800 microcomputer system from National Semiconductor represents a new branch on the 8080 family tree. It combines the best features of the 8080, the 8085 and the Z-80 with a new fabrication technique called poly CMOS. This proprietary new process (tagged "P²CMOS" by National) combines the speed of NMOS circuits with the low power consumption of CMOS.

Initial offerings in this product line include the NSC800 CPU, the NSC810 RAM-I/O-timer and the NSC830 ROM-I/O. All three are constructed using the same low-power P²CMOS. This chip set can be combined into a complete controller consisting of the three 40-pin packages, a crystal, one resistor and a capacitor.

This combination provides 2K bytes of ROM, 128 bytes of RAM and 32 bits of I/O that can be programmed to be either inputs or outputs. Power required can vary from 3 to 12 V dc, and the low power drain would enable full-speed operation for two days from a battery back-up system consisting of three D-size nicad batteries.

The familiar features of the new National microprocessor system should appeal to its users. The CPU uses the pin-out of the Intel 8085 and executes the entire Z-80 instruction set.

Rather than trying to establish a new dynasty of its own, National wisely copied the best features of earlier members of the 8080 family. This means that users of the earlier 8080, 8085 and Z-80 will be instantly able to design hardware and write software for the new microprocessor. Existing development systems, assemblers and PROM programmers can supply all the support required for this new system.

The NSC800 chip set and user's manual were not available as this was written, but the prospective user can gain valuable insight into techniques usable with the NSC800 by looking back at its predecessors. The reasoning behind National Semiconductor's decision to merge the pin configuration of the 8085 with the instruction set of the Z-80 unfolds as we look back at the history and evolution of the 8080 microprocessor family.

### The Roots: Intel's 8080

The 8008 microprocessor was relatively insignificant compared to its big brother, the 8080. Since its introduction, the 8080 and its descendants have become the standards of the microprocessor industry. If you don't want to accept that fact, just look at the relative amounts of available software and the number of different computer systems based on the 8080 family, although other micros may provide features lacking in the '80 family.

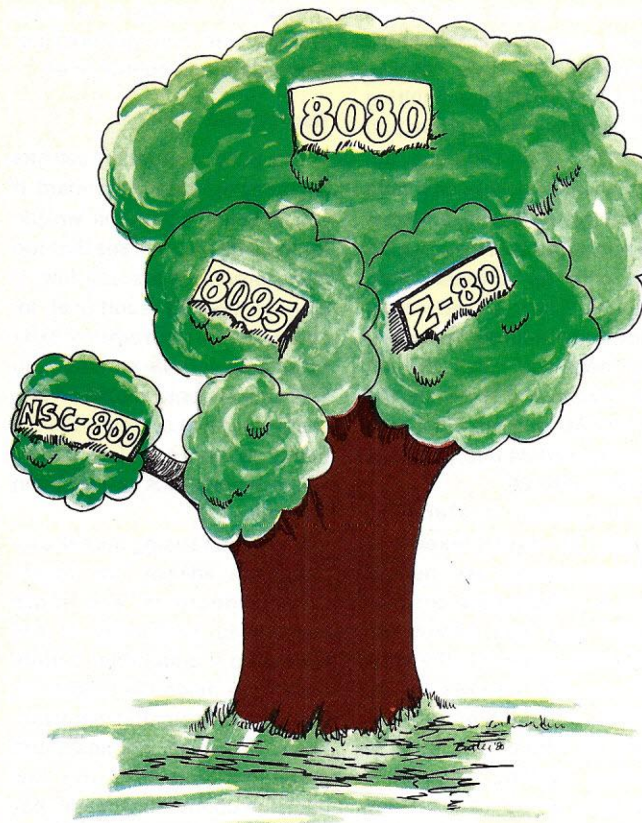
The 8080 was not the ultimate. To implement any 8080-based system, the 8080 central processor (CPU) has to be supported by a bi-phase, high-level clock generator; a system controller; and bus buffers. In spite of this, however, new products are still being introduced based on the 8080, even though its successors provide greatly expanded capability combined with reduced complexity.

To eliminate some of the 8080's support requirements, as well as add expanded capability, Zilog's Z-80 and the Intel 8085 have since appeared. These micros are upward compatible with the '80, executing the same instructions as the 8080 and adding instructions of their own.

### The Z-80 Instruction Mountain

The original designers of the 8080 used 244 out of the 256 possible combinations of eight bits as the basic instruction set. This left the door open a crack for the Zilog Z-80 designers to take advantage of the 12 unimplemented operation codes (op codes) to greatly expand the instruction set. With a couple of exceptions, this required a double level of instruction decoding.

The first eight-bit byte of the expanded instruction set tells the decoder that it will have to fetch an additional eight bits from the next memory location and decode that second byte to determine the desired operation. In this manner, 8-, 16-, 24- and 32-bit op codes have been implemented in the Z-80, at




the expense of additional memory fetches and execution time. Where speed is critical, as in servicing interrupts, the Z-80 executes two new single-byte instructions, the register exchanges EXX and EX AF.

The new multi-byte op codes implement functions that would have required many instruction and data fetches anyway, such as block data moves and bit manipulations, so there is still a net gain in efficiency in spite of the multiple-level decoding required by this technique.

While the major advantage of the Z-80 over the 8080 is the expansion of the instruction set, it also provides simplified interfacing to the outside world. Only a single-phase clock is required, and more usable bus control signals are provided by the CPU.

**The 8085**

Perhaps in answer to the advent of the Z-80, Intel next introduced the 8085 microprocessor. At first glance, the '85 almost seems to be a step in the wrong direction. Adding only two instructions to the 8080 op code set, it is a far cry from the Z-80's mountain of new op codes. The 8085 shines in the simplicity of implementing small systems, such as dedicated controllers.

The 8085 doesn't require an external clock generator. Simply connect a crystal across pins 1 and 2 to operate the '85. It even provides buffered clock output for other uses, such as supplying a bus clock or an input to a baud rate generator. If crystal timing accuracy is not required, substitute ten cents' worth of resistor and capacitor for the crystal.

For controlling the real world in real time, the '85 provides five different hardware interrupts accessible through individual pins and, in addition, can still support the multilevel priority interrupts that are implemented in 8080 and Z-80 systems (as well as 8085 systems) by the addition of a separate interrupt controller. If five levels of hardware interrupt are enough for a particular application, however, all are instantly available at separate pins on the 8085.

Additional simplification is provided by the inclusion in the '85 of "serial" input and output ports—in actuality, a one-bit input and a one-bit output port. In many applications, a "software UART" program can be written to make use of the serial I/O lines and emulate the hardware baud rate generator, crystal and UART chip usually required for communication with a terminal device.

Several of the functions of the 8080's system controller are implemented internally in the '85, providing signals for controlling the flow of data to and from memory and I/O devices. If more detailed CPU status information is required, status lines are also available, as was provided by the 8080's controller.

If the '85 is used with older 8080-style peripherals or memory, these status lines may have to be decoded to provide full 8080 compatibility. However, all newer memory and peripheral controllers are fully compatible with the '85 bus control signals, and the status output pins can be ignored in new designs. The one exception is shown in Fig. 1, where the status lines and a single NOR gate are used to light a "halt" indicator.

**The Multiplexed Bus**

To free enough pins in the standard 40-pin DIP package to provide for all these 8085 features, the eight-bit data and 16-bit address buses of the 8080 and Z-80 have been combined into a multiplexed eight-bit data-and-low-address bus and a non-multiplexed eight-bit high-address bus. An additional strobe comes out of the '85 on pin 30 to separate the data from the low-address bits on the multiplexed bus.

In any system—large or small—it is necessary to output a stable address before data

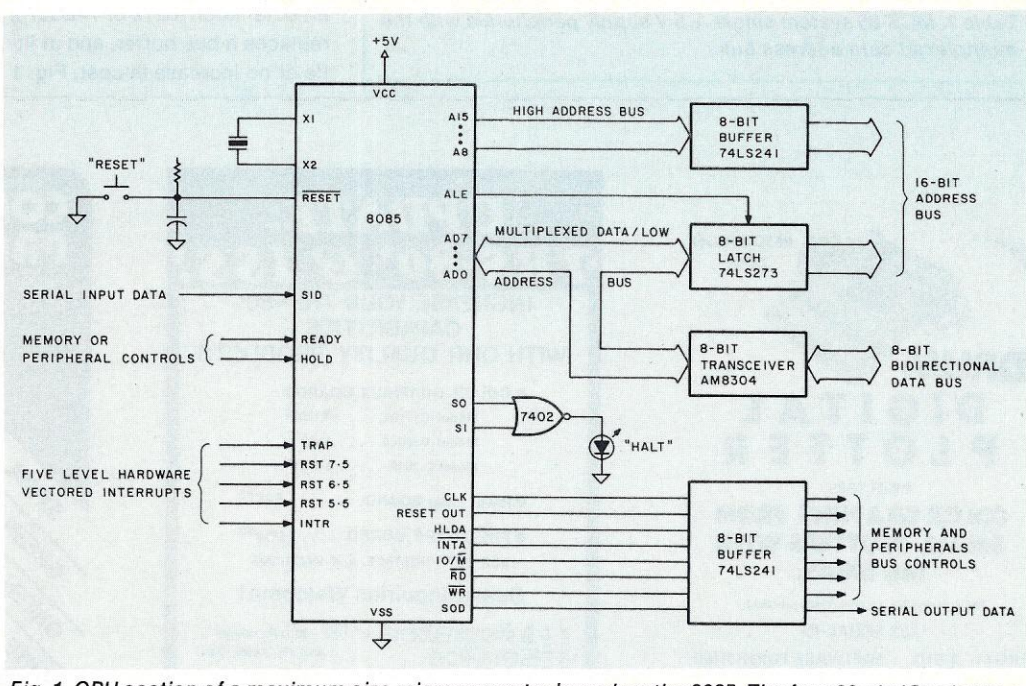



*Fig. 1. CPU section of a maximum-size microcomputer based on the 8085. The four 20-pin ICs shown on the right can provide all the address, data and control lines for systems with up to 64K bytes of memory and 256 I/O ports. Five vectored interrupts are available without the need for external interrupt controllers. All 40 pins on the 8085 package are used in the configuration.*

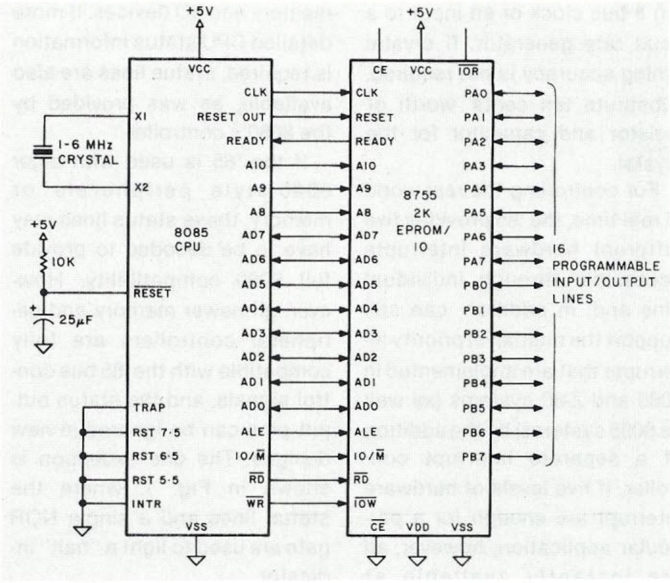



*Fig. 2. A minimum-size controller based on the 8085 microprocessor. The 8755 is a 2K byte ultraviolet erasable EPROM combined with two 8-bit input-output ports. For "RAM-less" applications, where CPU registers can supply sufficient read-write storage, the components shown here can comprise a complete controller operating from a single +5 V dc supply.*

| Device | Package | Features |
|---|---|---|
| 8155<br>8156 | 40-pin DIP | The 8155 and 8156 both provide 256 bytes of RAM, a 14-bit counter/timer, two eight-bit I/O ports and a six-bit I/O port. These two devices have complementary chip enable levels, permitting them to be used together without external address decoding. |
| 8185 | 18-pin DIP | A 1K byte static RAM in a small package, made possible by bus multiplexing. |
| 8355 | 40-pin DIP | 2K bytes of mask-programmable ROM; two eight-bit I/O ports. |
| 8755 | 40-pin DIP | 2K bytes of ultraviolet erasable EPROM; two eight-bit I/O ports. Pin compatible with the 8355, it permits program development for later inclusion in the 8355. |

*Table 1. MCS-85 system single +5 V supply peripherals with the multiplexed data/address bus.*

transfer can occur. This is true for read or write, memory or I/O operations. For this reason, there is no loss of time resulting from multiplexing address and data. The 16-bit address appears on the address bus (bits A8 through A15) and address-data bus (bits AD0 through AD7) along with an address latch enable (ALE) strobe used to save the low-order address bits in external hardware. With the address stable, and following the ALE, the AD bits are used as a bidirectional data bus, just as in the 8080 and Z-80.

In minimum systems, a number of multiplexed-bus peripheral circuits use the ALE signal to differentiate between address and data information sequentially applied to the same eight pins. Fig. 2 shows that no external components are required to implement such a minimum system. Compatible peripheral chips can provide RAM, ROM or EPROM memory, combined with counters, timers and I/O lines (see Table 1).

In larger systems—with more RAM and ROM memory—the address and data buses from the CPU have to be buffered in any case, so there is no increase in package count between the 8085 bus and that of the 8080 or Z-80. The only difference is that an octal latch (8212 or 74LS273) replaces a bus buffer, and at little or no increase in cost. Fig. 1 shows the nucleus of a large system, with fully buffered address, data and control lines.

### The NSC800

The 8080 instruction set has become the industry standard. If you examine programs written for the Z-80, you will see that too many programmers use few, if any, of the Z's additional instructions. Many reasons exist for this deplorable situation: no Z-80 assembler available on older development systems, the difficulty of converting to a new set of mnemonics or just plain lethargy.

If you aren't using the extra instructions anyway, the recent price reductions and more widespread availability of the 8085 and compatible peripherals have made the 8085 more attractive. Designing hardware based on the '85 is a snap, and managers just love the low package counts that result, but if only it contained the Z-80 op codes!

Sprouting a new trunk on the 8080 family tree, National Semiconductor has combined the super pin-out of the 8085 with Z-80 instructions. Now you can have the '85 pin-out, with its five instant hardware interrupts, serial I/O and single component clock generator, and the Z-80 instruction set executing at one million instructions a second, run by flashlight batteries! ■

# EXATRON STRINGY/FLOPPY Owners Association Newsletter

*Secretary, Fred Waters*

Readers will recall that last month in this newsletter we announced the advent of the Exatron Stringy/Floppy for the PET. Did YOU feel left out? Well. . . .

## THE APPLE ES/F IS HERE!!

The Exatron Stringy/Floppy—ES/F for short—is a mass storage subsystem for microcomputers. It does what an audio cassette machine does, but with very high reliability, and high speed. It does what a floppy disk subsystem does, a little slower and is half the cost. It's a way to store all your programs, both BASIC and machine language, quickly and surely, ready to load back into memory in a few seconds when needed. It has its own operating system, and is a superior way to handle the storage for your word processing tasks, software systems development projects, and data files for data processing routines. Seeing one of these remarkable microperipherals in action will convince you!

### WHERE DID IT COME FROM?

The Apple ES/F has a good pedigree. The Stringy/Floppy was introduced at the 2d West Coast Computer Faire in February 1978, for the S-100 bus. Later that year the ES/F for the SS-50 bus and the 6800 was perfected. In May 1979 the first ES/F for the TRS-80 was introduced at the 4th West Coast Computer Faire. Early in 1980 the ES/F with the RS-232 interface was on the market, followed by one for the PET. Exatron has been manufacturing handlers and test equipment for the electronics industry for many years, and brings those high standards to the development and manufacture of the Stringy/Floppy.

### WHAT DOES IT CONSIST OF?

The Apple ES/F consists of everything you need: a Drive Module, a Controller Card to insert in one of the Apple card slots, a flat ribbon cable connecting the two, and miniature tape cartridges called wafers. The Drive Module is housed in an attractive case to match your Apple; inside are the drive motor, the read and write tape heads, the read/write electronics, and operating controls for tape positioning and write protection. On the front face are the drive slot where you insert the wafer for SAVE or LOAD, and two indicator LEDs. The Controller Card contains the interface electronics for the Apple, and a ROM holding a bootstrap loader to load in the programs which make up the Stringy Operating System—SOS for short.

The Stringy/Floppy is physically integrated into your Apple II or Apple II Plus simply by inserting the ES/F Card into one of the card slots. It uses very little power, all of which is provided by the Apple bus.

The wafers are small tape cartridges 68mm x 40mm—two thirds the size of a business card—and 4.5mm thick. Inside is a continuous loop of digital quality tape in varying lengths from 5 feet to 75 feet providing different storage capacities. The wafer case is entirely enclosed except for a small slot where the drive capstan fits and another for contact with the tape head, for protection from handling and foreign particles. With each ES/F comes one wafer with SOS on tape, and a number of blank wafers for you to SAVE programs on.

The initial configuration has one Controller Card and one Drive Module. However the Controller will operate two Drives—all you do is connect the second unit. You can add as many Controller Cards as you have slots for—each one will allow two more drives.

### WHAT WILL IT DO?

When the SOS (Stringy Operating System) is loaded into Apple memory by the bootstrap loader in ROM, it integrates itself into your BASIC—whether Integer or Applesoft—and returns you to the BASIC prompt. You now have everything you had before in the way of BASIC commands, statements, and operation, with the addition of the complete SOS and all the ES/F commands. The ES/F commands will SAVE, LOAD, and RUN (load with autostart) programs in BASIC (either one or both); they will SAVE, LOAD, and RUN programs or data in binary (machine code); they will select either BASIC as desired when you have both in your Apple; and one command, CATALOG, displays a complete directory of the contents of a wafer.

The directory shows the position of each file on tape, the file type (Applesoft, Integer, or Binary), the starting address, the length of the file in bytes, and the file name. The SAVE commands include optional parameters for slot number, drive number, file position, and VERIFY. With VERIFY, SOS will SAVE the file with the necessary parameters, and then run the tape around the continuous loop and compare the file just SAVEd against memory. When you SAVE your programs this way, you KNOW you have them right on the wafer! File names can be up to 30 alphanumeric characters—you don't have to correlate numbers and names separately.

### HOW WELL DOES IT DO IT?

Here are some of the features, and what they will do for you. You're already familiar with the seemingly interminable delays in loading a program from audio cassette. The Apple ES/F saves and loads program material at 16,000 baud, or roughly 2K bytes per second. Tape speed is 10 inches per second. This means that you can save 10K bytes on the 5-foot wafer, and the ES/F will save and load this much material in six seconds!

What about reliability? Well, once you have certified a new wafer, it has a life expectancy of at least 10,000 passes. The error rate is so low that you may use the ES/F for weeks without ever running into a read or write error. There is a write-protect feature built into the ES/F to help you avoid operating errors.

Since the ES/F was designed from the ground up to digital standards for use with industrial quality equipment, you are not handcuffed to using audio equipment, audio materials, or audio standards for your Apple. There are no buttons, knobs, or switches to adjust when you save or load programs. The operations are all controlled by the software, and are highly reliable.

### WHY DO YOU NEED TWO DISKS?

Did you read Stutsman's article on page 84 of *Microcomputing* for August? It's titled "Why Do You Need Two Disks?" Jim Stutsman may never have tried out a Stringy/Floppy, but he puts forth some of the best arguments I've ever heard on why YOU need an ES/F. He wonders why (to paraphrase somewhat) we'd want to buy solutions to problems we've never had, and probably never will have. If you're not satisfied with audio cassette standards for your high-quality computer, but don't have the money to spend on one or two disks, check out the Exatron Stringy/Floppy. It meets Stutsman's suggested minimum standards for a good DOS, and will speak for itself when you try it out.

### PRICES AND ORDERING

The ES/F is assembled and tested at the factory, with a 30-day moneyback guarantee and a one year full warranty. For fastest delivery, phone in your credit card or COD order using the toll-free line below.

Base price for the TRS-80 ES/F, \$249.50 (ask about the Starter Kit); for the S-100 ES/F, \$289.50; Apple and PET with single drive is \$299.50, dual drive is \$449.50

Info packets at no charge; users manuals for the TRS-80 ES/F are available for \$3.00 for shipping.

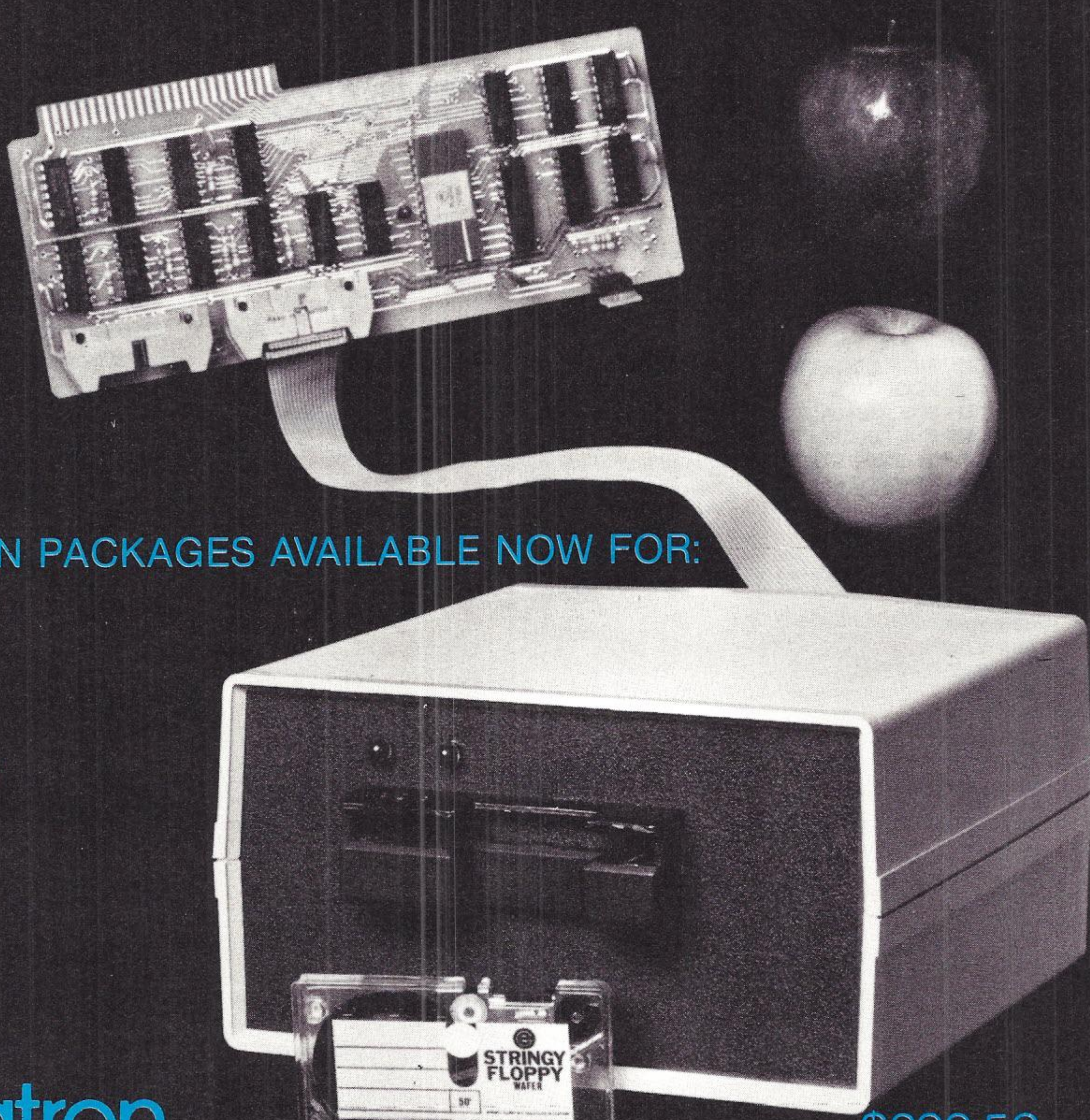
Handling is extra.

If you have any questions about these products, about Exatron or about ES/FOA call the Hot Line. Address letters to ES/FOA, 181 Commercial Avenue, Sunnyvale, CA 94086.

**Stringy/Floppy is a trademark of Exatron Corporation.**

| | |
|---|---|
| HOT LINE | 800-538-8559 |
| WITHIN CALIFORNIA | 408-737-7111 |

# FINALLY
## AN ALTERNATIVE TO DISKS
## THE EXATRON STRINGY FLOPPY
### (MASS STORAGE SUBSYSTEM)
### LOW COST · RELIABILITY · SPEED




INFORMATION PACKAGES AVAILABLE NOW FOR:
APPLE
PET
TRS-80
OSI
KIM/SYM/AIM
S-100
RS-232
STD-BUS
OEM

exatron

$299.50

## CALL OUR HOT LINE TODAY
## 800-538-8559

TO REQUEST AN INFORMATION PACKAGE

EXATRON, INC. ■ 181 COMMERCIAL STREET ■ SUNNYVALE, CA 94086

# Area Estimation

*It's simply a matter of BASIC geometry.*

```
100 REM MAIN PROGRAM — MUST SPECIFY X AND Y COORDINATES PRIOR TO
110 REM EACH CALL. SETTING FLAG F TO 1 RESETS ROUTINE. STOP INPUT BY
120 REM ENTERING NEGATIVE NUMBERS.
130 F=1
140 PRINT "ENTER THE X AND Y COORDINATES ? ";
150 INPUT X,Y
160 IF X<0 OR Y<0 THEN 190
170 GOSUB 500
180 GO TO 150
190 PRINT "THE AREA IS ";A
200 END
450 REM
460 REM AREA COMPUTATION SUBROUTINE. X1, Y1 SAVE COORDINATES FOR
470 REM NEXT CALL.  A IS AREA.  B KEEPS SUBTOTAL.  F=1 INDICATES TO
480 REM PROCESS ON FIRST CALL. X0, Y0 SAVE FIRST POINT COORDINATES.
490 REM F=0 RESETS FLAG.
500 IF F=1 THEN 570
510 A=B+X1*Y-Y1*X
520 X1=X
530 Y1=Y
540 B=A
550 A=-(A+X*Y0-X0*Y)*0.5
560 RETURN
570 X1=X
580 Y1=Y
590 B=0
600 A=0
610 X0=X
620 Y0=Y
630 F=0
640 RETURN
```

*Program A (for a Tektronix 4051).*

```
RUN
ENTER THE X AND Y COORDINATES ? 0 0
2 3
0 4
2.6666 4
4 6
5.3333 4
8 4
6 3
8 0
4 2
-1 -1
THE AREA IS 18.6667
```

*Program A sample run (corresponds to Fig. 1).*

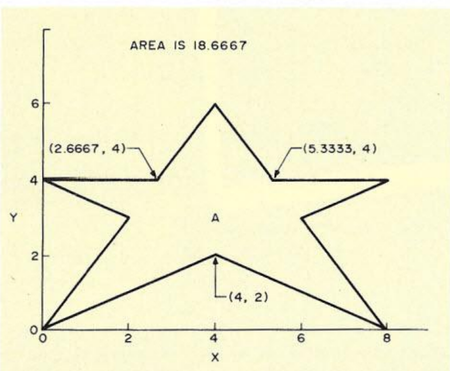



*Fig. 1. Example of irregular figure whose area is estimated by Program A.*


*Arnold W. Bragg*
*409 Cedar Hill Lane*
*Raleigh, NC 27609*


An educated guess is often the best we can do when calculating the area of complex and irregular geometric shapes. While the area can sometimes be estimated by subdividing the figure into a mosaic of rectangles, triangles and circles, this piecemeal technique is unsuitable for complex figures, and often lacks the precision required in sophisticated applications. A more precise computational tool is needed, one that will suit both the engineer and the hobbyist.

The area estimation software described here can calculate areas under function curves and areas of machine-generated figures, map sections, floor plans, survey plats and aerial photograph segments. Students can use the software routine to check geometry and integral calculus computations, or compute the surface area of butterfly wings and flower petals.

The software has been designed with three criteria in mind. First, the area computation algorithm must be a distinct module that does not require modification as the application changes. Second, the algorithm must be designed to run on machines having 4K of memory. Finally, the resulting area estimate must be mathematically correct. Compliance with these features guarantees the algorithm's suitability over a wide spectrum of users, machines and applications.

To estimate the area of any irregular shape we need only furnish the algorithm with the x- and y- coordinates of an arbitrary number of points lying on the figure's perimeter. The more data points we furnish, the better our estimate becomes. Because we are approximating curved segments with tiny straight line segments, the density of our data points should reflect the curvature of the perimeter.

At this stage we impose the algorithm's only restriction: We must traverse the figure in a clockwise manner. This guarantees that the area of the figure has the correct sign. The restriction assures us of mathematical correctness, which will be important later when we consider areas above and below an axis baseline.

```
RUN
ENTER INITIAL X VALUE ? 0
ENTER FINAL X VALUE ? 2
ENTER INCREMENT OF X ? 1
THE AREA IS 4

RUN
ENTER INITIAL X VALUE ? 0
ENTER FINAL X VALUE ? 2
ENTER INCREMENT OF X ? 0.1
THE AREA IS 3.34

RUN
ENTER INITIAL X VALUE ? 0
ENTER FINAL X VALUE ? 2
ENTER INCREMENT OF X ? 0.01
THE AREA IS 3.3334

RUN
ENTER INITIAL X VALUE ? 0
ENTER FINAL X VALUE ? 2
ENTER INCREMENT OF X ? 0.001
THE AREA IS 3.33333399999
```

*Program B sample run 1 (corresponds to Fig. 2).*

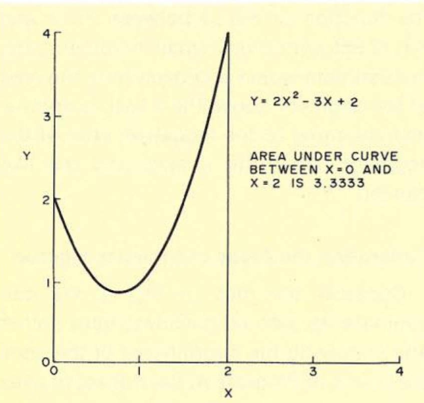



*Fig. 2. Example of bounded area estimated by sample run 1 of Program B.*

```
100 REM MAIN PROGRAM — FUNCTION PROCESSING — LINE 240 DEFINES FUNCTION
110 REM AND I, J ARE BOUNDS ON X, K IS X INCREMENT SIZE, X AND Y ARE
120 REM CALCULATED PRIOR TO EACH CALL.
130 PRINT "ENTER INITIAL X VALUE ? ";
140 INPUT I
150 PRINT "ENTER FINAL X VALUE ? ";
160 INPUT J
180 PRINT "ENTER INCREMENT OF X ? ";
190 INPUT K
200 F=1
210 X=I
220 Y=0
230 GOSUB 500
240 Y=2*X*X-3*X+2
250 GOSUB 500
260 X=X+K
270 IF X<=J THEN 240
280 X=J
290 Y=0
300 GOSUB 500
310 PRINT "THE AREA IS ";A
320 END
450 REM
460 REM AREA COMPUTATION SUBROUTINE. X1, Y1 SAVE COORDINATES FOR
470 REM NEXT CALL.  A IS AREA.  B KEEPS SUBTOTAL.  F=1 INDICATES TO
480 REM PROCESS ON FIRST CALL. X0, Y0 SAVE FIRST POINT COORDINATES.
490 REM F=0 RESETS FLAG.
500 IF F=1 THEN 570
510 A=B+X1*Y-Y1*X
520 X1=X
530 Y1=Y
540 B=A
550 A=-(A+X*Y0-X0*Y)*0.5
560 RETURN
570 X1=X
580 Y1=Y
590 B=0
600 A=0
610 X0=X
620 Y0=Y
630 F=0
640 RETURN
```

*Program B.*

```
RUN
ENTER INITIAL X VALUE ? 2
ENTER FINAL X VALUE ? 8
ENTER INCREMENT OF X ? 1
THE AREA IS -41.75

RUN
ENTER INITIAL X VALUE ? 2
ENTER FINAL X VALUE ? 8
ENTER INCREMENT OF X ? 0.1
THE AREA IS -41.9975

RUN
ENTER INITIAL X VALUE ? 2
ENTER FINAL X VALUE ? 8
ENTER INCREMENT OF X ? 0.01
THE AREA IS -41.999975

RUN
ENTER INITIAL X VALUE ? 2
ENTER FINAL X VALUE ? 8
ENTER INCREMENT OF X ? 0.001
THE AREA IS -41.9999997499
```

*Program B sample run 2 (corresponds to Fig. 3).*

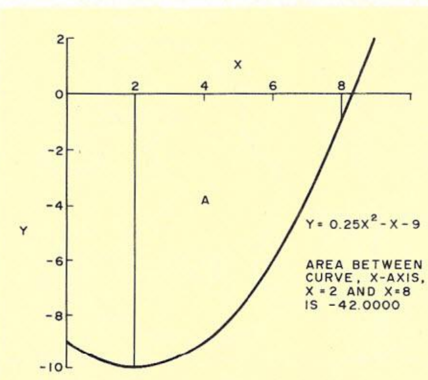



*Fig. 3. Example of negative bounded area estimated by sample run 2 of Program B.*

### The Algorithm

Because the area estimation algorithm is suited to different applications, it has been incorporated into a subroutine that can be appended to a user's BASIC program as needed. The subroutine requires only three values at each call: the data point's x-coordinate, its y-coordinate and a status variable, or flag, whose value must be set by the user prior to the first subroutine call.

The subroutine computes the area of a polygon whose vertices (corners) are the data points entered so far. For example, after three subroutine calls, the algorithm has calculated the area of the triangle whose vertices are the three data points supplied with each call. The area is zero after the first and second calls because polygons with one or two vertices enclose no area.

Program A illustrates how to calculate the area of the flattened star of Fig. 1. Each vertex (corner) of the star represents a data point supplied to the subroutine (lines 450–640) by an INPUT statement in the main program (lines 100–200). Our sign restriction forces us to traverse the star in a clockwise manner. Otherwise the area would be of the correct magnitude but would have a negative sign. We halt the routine by entering a negative X or Y value.

Variable F is the status flag, which is set to 1 in line 130 to indicate a first call. F is tested in line 500 for first call and reset in line 630 of the subroutine to zero for subsequent calls.

Variables X and Y represent the x- and y-coordinates of the most recently entered data point, and are supplied by the main program before each call.

Variables X0 and Y0 save the coordinates of the first data point for later processing. These values essentially allow us to "close the box."

Variables X1 and Y1 are the coordinates of the next-to-last data point entered. Variable B keeps a running subtotal of the area, and variable A is the area of the polygon defined by the data points.

### Calculating the Areas under Function Curves

Let's consider several examples. If we wish to compute the area under a function curve between two distinct points (Fig. 2) and we don't know the equation of the function we are dealing with, we use the approach outlined in Program A. We plot the function, estimate the x- and y-coordinates of a reasonable number of data points and supply these in sequence to the subroutine. Generally, the more points estimated, the better our area estimate becomes.

If we know the equation of the function, we need only a simple routine to specify the beginning and ending X values and to incrementally calculate and pass x- and y-coordinates to the subroutine. Consider the equation $Y = 2X^2 - 3X + 2$ between $X = 0$ and $X = 2$ (Fig. 2). Program B defines the main procedure and illustrates how the area estimate approaches the true area as the magnitude of the X increment is decreased. We must specify the starting point on the X axis ($X = 0$, $Y = 0$) and call the subroutine. Our program increases X by some increment, calculates the corresponding Y value, then passes these new coordinates to the subroutine. When X has been incremented to 2 (the right bound), we drop to the x-axis and define the last point ($X = 2$, $Y = 0$). We conclude by calling the subroutine for the last time and displaying the estimated area.

Lines 130-190 of Program B request the left and right bounds and the X increment. Line 240 defines the functional equation and line 270 tests for the right bound of X. Line 310 displays the resulting area estimate.

If a segment of the function lies below the x-axis, then the area estimate of that segment has a negative value. Consider the equation $Y = 0.25X^2 - X - 9$ between $X = 2$ and $X = 8$ (Fig. 3). Changing line 240 of Program B to Y = .25*X*X − X − 9 produces the estimates of Sample run 2. If we processed the function of Fig. 3 between $X = 2$ and $X = 12.955$ with a very small increment size, our estimate would approach zero; the area of the segment above the X axis is approximately equal to the (negative) area of the segment below the X axis, and the two cancel.

```
100 REM MAIN PROGRAM — LINES 140-200 COMPRISE CALIBRATION MODULE, LINES
110 REM 210-290 COMPUTE AREA OF SEGMENT, ADJUSTMENT FOR CALIBRATION
120 REM FACTOR IS MADE IN LINE 270.  R IS CALIBRATION UNIT AREA.
130 F=1
140 PRINT "ENTER THE X AND Y COORDINATES FOR CALIBRATION FIGURE ? ";
150 INPUT X,Y
160 IF X<0 OR Y<0 THEN 190
170 GOSUB 500
180 GO TO 150
190 R=A
200 PRINT "THE AREA OF THE CALIBRATION FIGURE IS ";A
210 F=1
220 PRINT "ENTER THE X AND Y COORDINATES ? ";
230 INPUT X,Y
240 IF X<0 OR Y<0 THEN 270
250 GOSUB 500
260 GO TO 230
270 A=A/R
280 PRINT "THE CALIBRATED AREA IS ";A; "UNITS"
290 END
470 REM NEXT CALL.  A IS AREA.  B KEEPS SUBTOTAL.  F=1 INDICATES TO
480 REM PROCESS ON FIRST CALL. X0, Y0 SAVE FIRST POINT COORDINATES.
490 REM F=0 RESETS FLAG.
500 IF F=1 THEN 570
510 A=B+X1*Y-Y1*X
520 X1=X
530 Y1=Y
540 B=A
550 A=-(A+X*Y0-X0*Y)*0.5
560 RETURN
570 X1=X
580 Y1=Y
590 B=0
600 A=0
610 X0=X
620 Y0=Y
630 F=0
640 RETURN
```

*Program C.*

### Calibrating the Areas of Irregular Figures

Consider the map in Fig. 4. We can estimate its area by selecting data points and supplying the coordinates of the sample points to Program A. But our set of axes has no grid as in Fig. 1. How can we estimate the X and Y coordinates? In what sort of units will our result be?

We obtain the coordinates of the data points by digitizing the outline of the figure. The digitization process can be accomplished by using a digitizing tablet or an optical scanner, or by overlaying the map with a sheet of ruled graphing paper

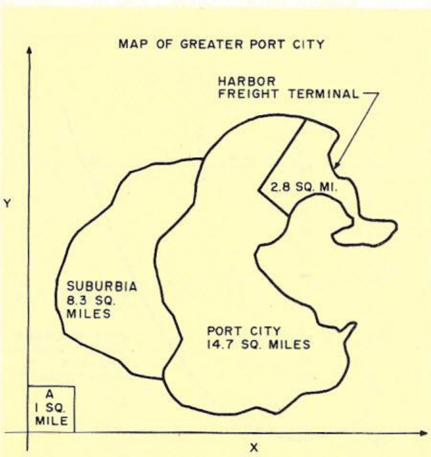



*Fig. 4. Example of calibrated area estimated by Program C.*

```
RUN
ENTER THE X AND Y COORDINATES FOR CALIBRATION FIGURE ? 0 0
0 5
5 5
5 0
-1 -1
THE AREA OF THE CALIBRATION FIGURE IS 25
ENTER THE X AND Y COORDINATES ? 5 10
4 11
3.5 12
. . .

10 6
9 6.5
6 8.7
-1 -1
THE CALIBRATED AREA IS 8.25017 UNITS
```

*Program C sample run (corresponds to Fig. 4).*

and reading the coordinates directly. Then, by defining a unit of known area (for example, box A of Fig. 4 = 1 square mile) and dividing our final result by this calibration factor, we guarantee that our results are independent of the coordinate scaling factor or grid size chosen, and are also expressed in terms of the units measured (in this case square miles). Program C illustrates how the calibration program estimates the area of the SUBURBIA subsector of Fig. 4. Notice that the calibration module uses the area subroutine to calculate correction factor R (line 130–200).

As indicated in Programs A, B and C, the area estimation subroutine is suited to several different applications, and requires only a small amount of memory for implementation. Applications differ only in the method by which data points are selected. The selection methods alter the calling routine; the subroutine never needs to be modified.

Program A is used when a convenient grid scale exists and data points with non-negative coordinates can be read directly from the scale, traversing clockwise. Program B is suited for numerical integration applications between specific bounds. Area segments below the x-axis display the correct sign. Program C is useful when the area estimate must be expressed in standard or non-standard units such as square inches or acres rather than in terms of the unit imposed by the grid scale.

In each case, our accuracy depends only on the number of sample data points we select to reasonably represent the figure and the precision with which the x- and y-coordinates of these data points are estimated. ■

# Thoughts on the SWTP Computer System

***The author takes a closer look at the 6809 and its Motorola BASIC, discusses multiprogramming and presents two interrupt-driven printer programs.***

*Peter A. Stark*
*PO Box 209*
*Mt. Kisco, NY 10549*

Since my comments on the 6809 appeared in the January issue, several readers have chided me for urging caution. Perhaps I wasn't clear enough in my comments. Here's what I intended to say.

I'm impressed by the 6809. It's about the best microprocessor around. The fact that so many other manufacturers are planning computers using it—including the TRS-80 Color Computer—is testimony to the fact that others think it's great too.

But as of now, its capabilities are not being utilized. 6809 software as of last January was simply reassembled 6800 code. While this software runs faster on the 6809, it doesn't take full advantage of the 6809's potential. Thus, rushing to convert to the 6809 just didn't make sense, especially since it wasn't definite whether the approaches of several different companies active in the software and hardware areas would be compatible. Hence, my caution and somewhat negative article.

In the meantime, there has been frantic activity to develop the software to take advantage of the 6809's capabilities. Moreover, what started out as bedlam—with each manufacturer heading off in a different direction—is now being coordinated, with most of the SS-50 bus manufacterers keeping in contact and exchanging hardware and software to make sure their products are compatible. That is a welcome sign.

#### Motorola's BASIC-09

One of the most important developments is a new BASIC from Motorola. As you remember, the 6800 received a bad and undeserved reputation for being slow simply because the first 6800 BASICs were slow (though excellent in other respects). It makes you wonder where the 6800 would be with respect to the 8080 and Z-80 had Motorola initially spent the money to develop an excellent BASIC, which they are now developing for the 6809.

For over a year, Motorola has had a contract with Microware to develop a new BASIC called BASIC-09. A preliminary version was on display at the Gimix booth at the West Coast Computer Faire in March and caused quite a stir. I've read the preliminary manual and hope to have a chance to try it out soon for a more complete report. But the manual is amazing.

BASIC-09, a structured language, borrows some of the better features of Pascal. Line numbers are optional; the program is broken up into sections called procedures, with each having a name. Procedures then call each other by name. Line numbers, if used, as well as variables, are local to the procedure, so other procedures can use the same line numbers or variables. If you're into structured programming, then you'll like this BASIC; if not, then just ignore procedures and write the whole program as one big procedure.

BASIC-09 has five types of variables—byte, integers (up to 32,767), real (nine-digit precision using 40-bit floating point binary), Boolean (true-false) and string (with string length up to 32,768 characters). You can define your own data formats by combining these five data types into larger data structures.

For BASIC old-timers, BASIC-09 has the standard GOTO statement; for structured programming aficionados who don't like GOTO, there are also
IF...THEN...ELSE
REPEAT...UNTIL
WHILE...DO
and other structures in Pascal.

BASIC-09 is a semi-compiler like the new TSC BASIC; that is, as each line is typed in, it is immediately checked for accuracy and translated into an internal code that makes later execution faster. When the program is listed, it is translated into plain BASIC and printed in an indented form for easy reading. BASIC-09 also includes an editor function that has extensive editing capabilities, including renumbering and string replacement. Variable names and procedure names can be any length, upper or lowercase.

For debugging, BASIC-09 allows tracing, which prints out each source line before it is executed, as well as the values of variables as they are assigned. At any point, the program can be suspended, and the variables can be read out or changed.

Microware has also developed an operating system called OS-IX (pronounced OS-nine). Although OS-IX is available with BASIC-09, it is also available separately from Microware. It is essentially a device-independent operating system that provides the interface between BASIC-09 or other languages and the hardware.

BASIC-09 will probably be available in ROM from Motorola as well as in disk-based versions directly from Microware.

For those users who want disk-based BASIC, Microware has signed a contract with Microsoft and will supply Microsoft BASIC 5.0, along with OS-IX, on disk. The advantage of this is that all the business software that now runs on other versions of Microsoft BASIC will also run with this BASIC. Present plans are to make this software compatible with both 5 inch and 8 inch floppy disks, as well as several different disk controllers.

At this point, it is still not definite as to what hardware will be required for what software. If you're happy with your 6800 system, stick with it until some more dust settles.

#### Time-Sharing and Multitasking

Most of us use our 6800 systems for just one job at a time. But the 6800 is versatile enough that it can execute several functions at one time. All of these require interrupts in one way or another.

The top eight locations in a 6800's memory, locations FFF8 through FFFF, contain four transfer vectors that correspond to the three interrupt inputs and reset as follows:
FFF8 and FFF9 – IRQ interrupt
FFFA and FFFB – SWI interrupt
FFFC and FFFD – NMI interrupt
FFFE and FFFF – Reset
(Reset is not really an interrupt, but it

behaves in a similar way.)

When the RESET switch is pressed, for instance, the 6800 stops whatever it is doing, fetches the vector that is stored in locations FFFE and FFFF in ROM and executes a jump to the address that the vector points to. For instance, in SWTBUG or MIKBUG, locations FFFE and FFFF contain the number E0D0. Hence, pushing RESET will force the computer to execute a jump to location E0D0, which is the beginning of the monitor.

Interrupts act a little differently. When one of the two interrupt inputs to the 6800 (either IRQ or NMI) is grounded, or when the program encounters an SWI instruction, the 6800 also completes its current instruction and then jumps to wherever the corresponding vector in ROM is pointing. However, before that jump it stores the contents of its internal registers in the stack. Later, an RTI instruction will fetch all of the information back off the stack and allow the 6800 to resume running the program that had been interrupted as though nothing happened.

The three interrupt methods have different uses. The SWI instruction is usually used for debugging (we have already seen in this series how it can be used with breakpoints and single-stepping). It can also be used for subroutine calls. For instance, GMXBUG uses the SWI to return to the monitor so that a system subroutine can be executed.

IRQ interrupts are usually used to allow I/O devices to call for help from the processor. An IRQ interrupt request (via a grounded IRQ pin) will be ignored by the 6800 if the I bit in the condition code register is a 1, so that important programs can turn on this bit and prevent IRQ interrupts from occurring.

This important feature is called masking. Certain kinds of programs—such as disk reads and writes, which have critical timing and would make errors if they were interrupted—can set the I bit and prevent themselves from being interrupted. Furthermore, once an interrupt occurs, further interrupts are usually masked to prevent some other problems.

NMI interrupts, on the other hand, are not controlled by the I bit in the condition code register; NMI stands for nonmaskable interrupt. When an NMI interrupt request occurs—when the NMI pin on the 6800 is grounded—an interrupt will occur, regardless of what else is going on.

Hence, NMI interrupts are usually reserved for important events, where the possible loss of data is not important. A typical case is a power-fail interrupt, where a sensor in the power supply generates an NMI interrupt when power fails, with the idea of giving the 6800 a few milliseconds—until power supply capacitors discharge, perhaps—to stop executing a program and either save data or at least go into a fail-safe mode before all power goes off.

Thus, for time-sharing or multitasking, we're usually concerned with the IRQ interrupt. IRQ interrupt requests can either come from an I/O device that asks for an interrupt when it needs the CPU or from an external timer that generates interrupts at fixed intervals.

Why would you want to generate these timed interrupts? There are several possible reasons:

1. An interrupt routine could keep count of these interrupts and use them to keep track of the time and date, which could then be stored in some memory locations and read out by other programs. Such an approach was described in an article by Richard Parry in the January 1980 issue of *Kilobaud Microcomputing* on page 150. (Richard's scheme was a bit indirect: interrupts did not go to the 6800 directly, but went through a PIA first. But that's not important.)

2. A timed interrupt could be used to time external outputs. For instance, if we wanted

to output serial five-level code out of one of the outputs of a PIA at 45 baud, then an interrupt occurring every 1/45 of a second could be used to time the outputs.

3. When we want to run several programs in a round-robin fashion, giving each a certain amount of time, an interrupt timer can cause the transfer of control from one program to the next. This is essentially the idea behind time-sharing. There are several 6800 products designed for running multiple programs in this fashion.

### Microware's RT-68

Our discussion of monitors has already covered some of the features of the RT-68; let's look at just its multitasking operation.

With the RT-68, CPU time is allocated to each of up to 16 programs (called tasks) in a round-robin fashion, so that these tasks appear to run concurrently. These tasks may be independent or may depend on each other and share data.

RT-68 keeps track of these by maintaining a task status table in locations A050 through A07F. For each of the 16 possible tasks, the table contains three bytes of information. The first of these is a TSB, or task status byte, while the other two contain a TSP, or task stack pointer.

Tasks are started and stopped by interrupts; since each interrupt stores register contents in the stack, each of the 16 tasks has to maintain its own stack area to keep this data separate from all other tasks. Thus, the task stack pointer in the table points to the stack used by each task. Keep in mind that of the 16 possible tasks, only one task is actually running at any one time; task stack pointers for the other 15 are constantly kept in the table.

Tasks can have one of three states:

1. Currently executing.
2. Active, that is, not currently running, but ready to run as soon as their turn comes.
3. Inactive, that is, not currently running and also not ready to run.

Although RT-68 can handle up to 16 tasks, in most cases a much smaller number of tasks will be active. There could, for instance, be just two active tasks and 14 inactive ones.

The task status byte is divided into three sections:

1. Bit 7 indicates whether the task is active (1) or inactive (0).
2. Bits 6 through 3 indicate how long that task is to run before being stopped and control turned over to another task. This time is measured in interrupts received from an interrupt timer; each interrupt is called a tick, and a task may be assigned from one to 15 ticks.
3. Bits 2 through 0 indicate a task's priority. Each task can be assigned a priority from 0 to 7. Higher-priority tasks get preference over low-priority tasks; in fact, a low-priority task will not run at all as long as there are active higher-priority tasks. There is also a system priority that RT-68 keeps for itself; tasks having a lower priority also don't get a chance to run.

RT-68 uses an external clock timer, which generates an interrupt at a fixed interval, such as every 1/60 or 1/100 second. This is coupled to a PIA on port 1, and through that to the NMI line. Thus, the clock generates an NMI interrupt at fixed intervals.

Normally, NMI interrupt requests cannot be masked, but RT-68 channels them through a PIA first so they can be controlled at the PIA. This converts them into a maskable form and also allows testing to see where an interrupt came from. Other interrupts can also be connected to the NMI or IRQ inputs. RT-68 can tell the difference between a clock interrupt and a user NMI interrupt by checking to see whether it came from the PIA on port 1.

### Other Available Programs

Several programs for multi-programming with the MP-T timer are available from Ed

Joyce, 4603 Lyceum Drive, San Antonio, TX 78229. His scheme also uses a task status table and some similar ideas.

If you have access to old copies of *Electronic Design* magazine, a short note by David Johnson in the February 15, 1978, issue describes a simple approach to do the same thing. In this case, switching back and forth between tasks is initiated by each task periodically suspending itself by calling the monitor. This approach is simpler but has the disadvantage that a program that develops a bug may not release control.

**SWTP Multi-User Board**

The approaches I've discussed so far are for running a number of completely separate programs; the multi-user board (MUB), on the other hand, is set up for running one program with several users. It's intended to allow up to four users to run programs in either the BASIC or Pilot language. Either way there is only one BASIC or Pilot interpreter in memory, and the four users take turns using it to interpret their own user programs.

The MUB board occupies memory addresses 0000 through 0FFF, or the lowest 4K. Thus, only addresses from 1000 and up are available for user RAM. SWTP states that the remaining memory from 1000 to 7FFF must be built up out of 4K and 8K boards and that the 32K memory board cannot be used with the MUB. This is not so, as we shall see later.

In any case, the MUB uses up addresses from 0000 through 0FFF, or 4K. On the board are four 2114 static RAMs, which provide 2K bytes of memory, which is split up into four 512-byte sections, one section for each user. As in a typical timesharing system, each user gets a small slice of time, typically about five milliseconds, during which his program runs and the other users' programs are dormant. During this time, his 512 bytes occupy addresses 0000 through 01FF, while the other three 512-byte memories are disconnected.

The MUB has a page register at address 0C00 that selects which of the four 512-byte memories is enabled at any one time. The variables, stack and various constants for each user are stored in these 512 bytes and swapped in and out by hardware when a different number is written into this page register to select a different user.

An interrupt timer on the board generates an IRQ interrupt about 200 times per second (about five milliseconds apart), so no external timer board is required. These interrupts are generated by a 555 timer, which is disabled when the system is first started but starts working when a read from location 0800 is executed. Once the clock starts, it continues running until power is turned off or the reset button is pressed.

The hardware of this board is fairly simple; it's the software that makes the whole operation useful. Normally supplied with the MUB is a 4K Micro BASIC Plus, which allows up to four users at the same time. This is an integer BASIC without strings that appears similar to the Micro BASIC offered by TSC. In fact, the multi-user board and multi-user BASIC first appeared in some TSC ads a few years ago.

Also available are 8K cassette and disk-based BASICs from SWTP. Using the information in the fifth installment of this series (September 1979), you could modify SWTP 8K BASIC, version 2.0, to work in this mode too.

Of special interest to teachers, though, may be the multi-user Pilot also available from SWTP and Micropi. Pilot is a programmed instruction language specially set up for easy programming of question-and-answer-type programs.

Basically, Pilot commands consist of one or two letters followed by a colon (T: or M:), followed, in turn, by some text, for example:

```
T: What's your name?
A:
M: Pete, Peter
YT: I like that name.
NT: Too bad
```

This program says to type (T:) the question, "What's your name?" wait for an answer (A:) and match that answer (M:) against either Pete or Peter. If it matches (YT: means type if yes), print one message; if it does not match (NT: means type if no), print another.

Full Pilot allows branching and computation, but the convenient feature is that it also will allow partial matches of answers or even allow alternative answers. For example, it will accept an answer even if it is misspelled or will accept a numeric answer to a problem even if it is off by a slight amount. Though BASIC could also be used for this type of programming, it takes a lot of work in BASIC to allow for alternative or partial answers.

**MUB Modifications**

Since the multi-user board occupies the lower 4K of memory but only has 512 bytes of actual memory accessible at any one time, when it is plugged in, the system will not run any other programs that require the lower 4K.

Moreover, SWTP states that the MUB board cannot be used with the 16K or 32K dynamic memory board. Thus, to get a full 28K of memory (from 1000 to 7FFF) you need three 8K boards and a 4K board. When you unplug the MUB board to run normal programs, you then need one more memory board to supply the lower 4K. Thus, to use the system you need five memory boards, a lot of power and almost $1000 (at list price).

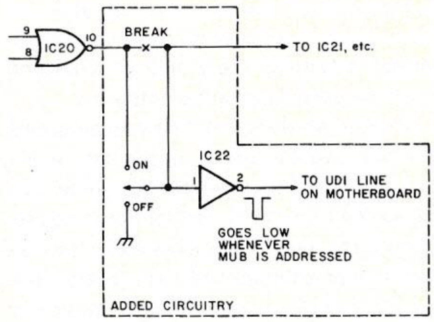



*Fig. 1. Changes to multi-user board.*

With a simple modification, the MUB will work just fine with the 32K memory board. Moreover, this modification includes a switch to enable or disable the MUB, so that the system can be used for standard single-user programs without having to plug and unplug boards. As with some other information in previous installments, this modification comes from a knowledgeable SWTP dealer, Lehigh Computer Works, 1132-2 Tilghman Street, Allentown, PA 18102. Tom Quay of Lehigh has tried this one out and it works well. I'll only outline the mod here; complete details can be obtained from Lehigh.

The modification simply disables the 32K memory board whenever the MUB is addressed. The only specific requirement is that the memory board contain the full 32K of memory; this is really of no consequence because for multi-user applications you'd probably want this much memory anyway. The only parts required for the modification are some wire, an SPDT switch and a 2.2k resistor.

The switch is optional, but it does allow the MUB to be switched out, so that the full 32K of memory becomes usable for normal program use. This modification also uses the UD1 line on the motherboard; if your system is already using this line for some other purpose, then you'll have to find some other way to get a signal from the MUB to the memory board (such as a direct jumper).

Fig. 1 shows the changes required on the MUB board. The connection from IC20, pin 10, to the rest of the board must be broken by cutting a printed circuit land or simply by bending pin 10 up and out of its socket, if an IC socket is used. The SPDT switch should be mounted on the upper left corner of the board and connected as shown in Fig. 1. IC22 has an unused inverter, which feeds the UD1 line on the bus. This signal goes low whenever the MUB is addressed.

Fig. 2 shows the changes to the 32K memory board. Jumpers E1 and E2 should be removed and a 2.2k resistor connected instead of E2. This would therefore permanently enable the board for the full 32K of memory from 0000 to 7FFF. But a wire from UD1 up to pin 2 of U42 as shown

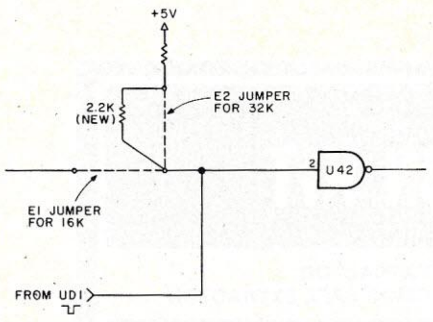



*Fig. 2. Changes to 32K memory board.*

disables the memory board whenever the MUB is addressed and sends out a low pulse on the UD1 line.

When the system is not being used with the multi-user software, the switch should be off. To run multi-user BASIC or Pilot, turn the switch on. The MUB can stay plugged in all the time.

### Another Way

If you are interested in multi-programming, keep in mind a comment I made several months ago while discussing expansion of the SWTP 32K board to 64K. If the board is configured as two separate 32K sections, switched by a bit coming from a PIA, then you can do what the MUB does, but on a larger scale. Instead of just switching 512 bytes in and out, you can switch 32K and provide full memory for each of two users. In one 32K segment, one user could do an assembly or run BASIC, while the other could do something completely different in his 32K.

### Printer Interrupts

One down-to-earth application of interrupts is to drive a printer. This is only practical if you keep a memory buffer filled with characters to be printed, while an interrupt routine empties it at the printer's own pace. In this way, the program can run ahead of the printer and pause only when the buffer becomes filled up.

In many programs, especially in BASIC, where the printer runs in bursts, this may speed up overall running time by as much as a factor of 2, because the printer can now run continuously instead of starting and stopping. The same is true of the program that drives it.

Applied Microcomputer Systems (Box 68, Silver Lake, NH 03875) has been offering an interrupt-driven parallel printer driver for some time. It is intended primarily for the SWTP PR-40 printer.

However, many of the newer printers, such as the Anadex, have a built-in buffer of 1K or more. These printers don't really need much more buffering and can be driven directly from a parallel port without any fancy programming.

A buffer would really be useful to drive a serial printer, especially a slow one. For instance, a 1K buffer on a 110-baud teleprinter would hold 100 seconds of output, allowing a program to get 100 seconds ahead of the printer This can be most impressive if you list a program on the printer, and can go on to something else while the printer is still listing.

The programming needed to do this is easy to use. Included in this article are two versions of such an interrupt-driven printer driver. Both require an MP-S port on port 1, having a jumper in the IRQ position to enable interrupts.

Listing 1 shows a simple routine that provides only a 256-byte buffer and is intended to sit in a high RAM memory address such as A800. It essentially consists of three parts.

An INITLZ routine must be called before any output to the printer driver is performed, and it sets up some pointers and flags, puts the address of the interrupt service subroutine (ISS) into the IRQ vector at A000 and returns to the calling program.

The routine uses a 256-byte buffer in memory, and two pointers are used to keep track of buffer locations. INPTR always points to the next empty location in the buffer that can be filled with an incoming character, while OUTPTR always points to the next character to be printed. The buffer is looped back on itself, so that when the last location is filled or emptied, the next location to be filled or emptied is at the beginning again. It resembles a never-ending photo slide tray, with the pointer continuously cycling through the buffer.

Whenever INPTR and OUTPTR are the same, the buffer is empty; whenever OUTPTR is equal to INPTR + 1, the buffer is full. These tests are used by the program to keep track of what is going on.

BUSYFL is a flag that indicates whether the ACIA is busy. Each time a character is sent out to the ACIA, BUSYFL is set to 1. It stays at 1 until the ACIA finishes printing and generates an IRQ interrupt to ask the processor for another character, at which time it goes back to 0. Thus, 1 in BUSYFL means that the ACIA has started printing a character, but hasn't yet finished it.

The second part of the program is the INPUT routine, which is designed as a replacement for INEEE and is needed because INEEE in most monitors reconfigures the ACIA on port 1 to turn off interrupts. This would stop all printing if allowed, so INPUT does the same thing, but without the initialization.

The main part of the program is the OUTPUT routine, which replaces OUTEEE. This routine was designed for use with standard programs such as BASIC, and so OUTPUT has some features that allow it to complete printing if BASIC returns to the monitor. (The monitor's calling INEEE would otherwise stop printing even if there was still some data in the buffer, because INEEE normally initializes the ACIA to turn off its interrupt system.)

This is done by putting into the calling program a print of a control-F at the end of output; in BASIC, this is done with PRINT CHR$(6). After this, the OUTPUT driver waits until the buffer is empty before returning to the calling program. This prevents the loss of the material in the buffer. Because of the way INPTR and OUTPTR are handled in this routine, this program cannot be placed in ROM.

Listing 2, on the other hand, shows another version of this program. This version is in ROM and is part of the HUMBUG monitor I described in the September 1980 *Microcomputing.*

In this version, the buffer is a 1K RAM area at D400 to D7FF. Because ten bits are now needed to point to a specific location in the buffer (as opposed to eight bits when the buffer was just 256 bytes long), INPTR and OUTPTR are now two bytes instead of one, and this complicates the programming. The INITLZ routine starts with a ten millisecond time delay to permit the ACIA on port 0 to finish any current operation and then clears INPTR, OUTPTR, POSTAT (the flag used by FCROM in the main monitor to steer non-interrupt output to port 0) and BUSYFL. Then it places the address of the ISS routine in the IRQ location of monitor RAM (at A000), puts a nonzero number into IRQON to indicate that initialization has been done and exits.

Just in case non-monitor programs want to use this routine, entry point IQPRNT is included just after the initialization routine. This entry is used by all such programs and automatically initializes whenever a control-E (hex 05) is printed. Otherwise, it falls through to OUTPUT.

### Flex

Flex 2.0 uses SWI interrupts to handle its printer spooling feature. It changes the SWIJMP vector at A012 and does not restore it when done. Be careful if you try to debug a machine-language program with breakpoints after using Flex 2.0, since breakpoints will now return to Flex 2.0 instead of to your monitor.

Converting programs from miniFlex to Flex 2.0 or back is generally not too difficult. Basically, any machine-language program that uses disk operations maintains a file control block, or FCB, for every open disk file. In miniFlex, this FCB contains a total of 192 bytes—64 bytes of status control info, followed by 128 bytes for the contents of one disk sector. Since Flex 2.0 has 256 bytes per sector, its FCB is 320 bytes long (64 + 256). The organization of the data in

Listing 1. Self-contained serial printer driver.

```
                        NAM   SPEEDUP

                 * INPTR POINTS TO THE NEXT EMPTY LOCATION IN BUFFER
                 * OUTPTR POINTS TO THE NEXT LOCATION TO BE OUTPUT
                 *    IF INPTR=OUTPTR, BUFFER IS EMPTY
                 *    IF OUTPTR=INPTR+1 BUFFER IS FULL
                 * BUSYFL INDICATES WHETHER ACIA IS BUSY
                 *    BUSYFL=1 MEANS ACIA HAS BEEN STARTED ON OUTPUT,
                 *             BUT INTERRUPT INDICATING COMPLETION HAS
                 *             NOT YET ARRIVED

                 * MONITOR AND I/O EQUATES

  (E1D1)         OUTEEE EQU   $E1D1
  (E07E)         PDATA  EQU   $E07E
  (E0D0)         MONITR EQU   $E0D0
  (8004)         ACIACR EQU   $8004
  (8005)         ACIADR EQU   ACIACR+1
  (A000)         IRQPTR EQU   $A000

  (A800)                ORG   $A800

                 * TRANSFER VECTORS INTO SPEEDUP
A800 7E A82E     VECTOU JMP   OUTPUT
A803 7E A820     VECTIN JMP   INPUT
A806 7E A809     VECTIZ JMP   INITLZ

                 * INITIALIZE ROUTINE
A809 B6 8004     INITLZ LDA A ACIACR    IN CASE ACIA IS BUSY, WAIT
A80C 47                 ASR A           FOR CURRENT OPERATION TO COMPLETE
A80D 47                 ASR A
A80E 24 F9              BCC   INITLZ
A810 7F A86B            CLR   INPTR     INITIALIZE INPUT AND OUTPUT PTRS
A813 7F A890            CLR   OUTPTR
A816 7F A8F4            CLR   BUSYFL    ACIA NOT BUSY
A819 CE A876            LDX   #ISS      POINTER TO ISS ROUTINE
A81C FF A000            STX   IRQPTR    INSERT IRQ TRANSFER ADDRESS
A81F 39                 RTS             AND RETURN

                 * INPUT FROM TERMINAL ROUTINE (NO INTERRUPTS USED)
A820 B6 8004     INPUT  LDA A ACIACR    WAIT FOR CHARACTER
A823 47                 ASR A
A824 24 FA              BCC   INPUT
A826 B6 8005            LDA A ACIADR    GET THE CHARACTER
A829 8D 03              BSR   OUTPUT    ECHO IT
A82B 84 7F              AND A #$7F      STRIP OFF PARITY BIT
A82D 39                 RTS

                 * OUTPUT TO TERMINAL ROUTINE
A82E 81 06       OUTPUT CMP A #$06      END OF DATA?
A830 27 15              BEQ   DONE      YES
A832 01                 NOP
A833 0F                 SEI             DISABLE INTERRUPT SYSTEM
A834 7D A8F4            TST   BUSYFL    IS ACIA BUSY?
A837 26 1B              BNE   BUSY      YES
A839 B7 8005     FREE   STA A ACIADR    NO; ACIA IS FREE SO LET'S OUTPUT
A83C 7C A8F4            INC   BUSYFL    TURN ON BUSY FLAG
A83F 86 35              LDA A #$35      ENABLE ACIA TRANSMITTER FOR IRQ
A841 B7 8004            STA A ACIACR
A844 01                 NOP
A845 0E                 CLI             ENABLE INTERRUPTS
A846 39                 RTS             AND RETURN TO MAIN PROGRAM

                 * IF DONE, CLEAN UP AND RETURN
A847 7D A8F4     DONE   TST   BUSYFL    WAIT FOR BUFFER TO EMPTY
A84A 26 FB              BNE   DONE
A84C 86 15              LDA A #$15      DISABLE ACIA TRANSMITTER INTERRUPTS
A84E B7 8004            STA A ACIACR
A851 01                 NOP
A852 0F                 SEI             DISABLE INTERRUPTS
A853 39                 RTS

                 * ACIA IS BUSY, SO STORE IN BUFFER
A854 37          BUSY   PSH B
A855 F6 A86B     BUSY1  LDA B INPTR     IS BUFFER FULL?
A858 5C                 INC B
A859 F1 A890            CMP B OUTPTR
A85C 26 04              BNE   NOTFUL
                 * FULL, SO WAIT UNTIL SOME SPACE IS AVAILABLE
A85E 01                 NOP
A85F 0E                 CLI             ENABLE INTERRUPTS
A860 20 F3              BRA   BUSY1     GO BACK AND CHECK AGAIN
                 * NOT FULL, SO STORE CHARACTER IN BUFFER
A862 01          NOTFUL NOP
A863 0F                 SEI             DISABLE INTERRUPTS IF ENABLED ABOVE
A864 FF A8F2            STX   SAVEX     SAVE INDEX REGISTER
A867 CE A8F5            LDX   #BUFFER
  (A86B)         INPTR  EQU   *+1       CAUTION - NOT PROMMABLE
A86A A7 00              STA A 0,X       STORE INTO NEXT EMPTY BUFFER LOC
A86C 7C A86B            INC   INPTR     INCREMENT IN POINTER
A86F 01                 NOP
A870 0E                 CLI             ENABLE INTERRUPTS
A871 33                 PUL B           RESTORE B AND INDEX
A872 FE A8F2            LDX   SAVEX
A875 39                 RTS             RETURN

                 *INTERRUPT SERVICE ROUTINE
A876 B6 8004     ISS    LDA A ACIACR    DID SOMETHING ELSE INTERRUPT?
A879 2A 20              BPL   ERROR     IF YES
                 * OK - INTERRUPT CAME FROM ACIA
```

```
A87B 7F A8F4          CLR   BUSYFL     ACIA NO LONGER BUSY.
A87E B6 A86B          LDA A INPTR      IS THE BUFFER EMPTY?
A881 B1 A890          CMP A OUTPTR
A884 26 06            BNE   NOTEMT     NO
A886 86 15            LDA A #$15       DISABLE ACIA TRANSMITTER INTERRUPTS
A888 B7 8004          STA A ACIACR
A88B 3B               RTI              RETURN
                * BUFFER NOT EMPTY, SO OUTPUT THE NEXT CHARACTER
A88C CE A8F5    NOTEMT LDX  #BUFFER
  (A890)        OUTPTR EQU  *+1        CAUTION - NOT PROMMABLE
A88F A6 00            LDA A 0,X        GET NEXT CHARACTER
A891 B7 8005          STA A ACIADR     OUTPUT IT
A894 7C A890          INC   OUTPTR     INCREMENT OUTPUT POINTER
A897 7C A8F4          INC   BUSYFL     ACIA IS BUSY AGAIN
A89A 3B               RTI              RETURN TO PROGRAM

                * ERROR ROUTINE - INTERRUPT NOT CAUSED BY ACIA
A89B 86 15      ERROR LDA A #$15       DISABLE ACIA INTERRUPTS
A89D B7 8004          STA A ACIACR
A8A0 B6 A86B    ERROR1 LDA A INPTR     SEE IF BUFFER IS EMPTY
A8A3 B1 A890          CMP A OUTPTR
A8A6 27 13            BEQ   EMPTY      YES
A8A8 B6 A890          LDA A OUTPTR     NO, SO EMPTY IT BEFORE QUITTING
A8AB B7 A8B2          STA A GETCHR
A8AE CE A8F5          LDX   #BUFFER
  (A8B2)        GETCHR EQU  *+1        CAUTION - NOT PROMMABLE
A8B1 A6 00            LDA A 0,X        GET NEXT CHARACTER
A8B3 BD E1D1          JSR   OUTEEE     PRINT IT
A8B6 7C A890          INC   OUTPTR     INCREMENT POINTER
A8B9 20 E5            BRA   ERROR1     GO BACK AND TEST AGAIN
                *BUFFER EMPTY, SO PRINT ERROR MESSAGE AND QUIT
A8BB CE A8C4    EMPTY LDX   #ERRMSG
A8BE BD E07E          JSR   PDATA      PRINT ERROR MESSAGE
A8C1 7E E0D0          JMP   MONITR     RETURN TO MONITOR
A8C4 0D         ERRMSG FCB  $D,$A,0,0,0,0
A8CA 45               FCC   /ERROR - INTERRUPT NOT CAUSED BY ACIA   /
A8F1 04               FCB   $04

                * DATA ETC.
A8F2            SAVEX RMB   2          INDEX STORAGE
A8F4            BUSYFL RMB  1          ACIA BUSY FLAG
A8F5            BUFFER RMB  256        256-BYTE BUFFER
                      END
```

```
                * PORT 0 PRINTER INTERRUPT DRIVERS

                * INPTR POINTS TO THE NEXT EMPTY LOCATION IN BUFFER
                * OUTPTR POINTS TO THE NEXT LOCATION TO BE OUTPUT
                *     IF INPTR=OUTPTR, BUFFER IS EMPTY
                *     IF OUTPTR=INPTR+1 BUFFER IS FULL
                * BUSYFL INDICATES WHETHER ACIA IS BUSY
                *     BUSYFL=1 MEANS ACIA HAS BEEN STARTED ON OUTPUT,
                *              BUT INTERRUPT INDICATING COMPLETION HAS
                *              NOT YET ARRIVED

                * INITIALIZE ROUTINE
ECCF FF D04A    INITLZ STX  SAVEDX
ECD2 CE 04E2          LDX   #$04E2
ECD5 09         INWAIT DEX             WAIT ABOUT 10 MILLISEC
ECD6 26 FD            BNE   INWAIT     FOR ACIA TO FINISH IF BUSY
ECD8 CE 0000          LDX   #0
ECDB FF D051          STX   INPTR
ECDE FF D053          STX   OUTPTR     RESET INPUT AND OUTPUT POINTERS
ECE1 7F D000          CLR   POSTAT     TURN OFF NORMAL PORT 0 OUTPUT
ECE4 7F D050          CLR   BUSYFL     ACIA NOT BUSY
ECE7 CE ED60          LDX   #ISS       RESET IRQ POINTER ADDRESS TO ISS
ECEA FF A000          STX   IRQ
ECED 7F D05A          CLR   IRQON
ECF0 7C D05A          INC   IRQON      TURN ON IRQ IN PROGRESS FLAG
ECF3 FE D04A          LDX   SAVEDX
ECF6 39               RTS              AND RETURN

                * NON-MONITOR ENTRY POINT

ECF7 81 05      IOPRNT CMP A #$05      ON CONTROL-E, GO INITIALIZE
ECF9 27 D4            BEQ   INITLZ     OTHERWISE FALL THROUGH TO OUTPUT

                * OUTPUT TO TERMINAL ROUTINE
ECFB 81 06      OUTPUT CMP A #$06      END OF DATA?
ECFD 27 15            BEQ   DONE       YES
ECFF 01               NOP              NO
ED00 0F               SEI              DISABLE INTERRUPT SYSTEM
ED01 7D D050          TST   BUSYFL     IS ACIA BUSY?
ED04 26 1E            BNE   BUSY       YES
ED06 B7 8001    FREE  STA A ACIADR     NO; ACIA IS FREE SO LET'S OUTPUT
ED09 7C D050          INC   BUSYFL     TURN ON BUSY FLAG
ED0C 86 35            LDA A #$35       ENABLE ACIA TRANSMITTER FOR IRQ
ED0E B7 8000          STA A ACIACR
ED11 01               NOP
ED12 0E               CLI              ENABLE INTERRUPTS
ED13 39               RTS              AND RETURN TO MAIN PROGRAM

                * IF DONE, CLEAN UP AND RETURN
ED14 7D D050    DONE  TST   BUSYFL     WAIT FOR BUFFER TO EMPTY
ED17 26 FB            BNE   DONE
ED19 86 15            LDA A #$15       DISABLE ACIA TRANSMITTER INTERRUPTS
ED1B B7 8000          STA A ACIACR
ED1E 01               NOP
ED1F 0F               SEI              DISABLE INTERRUPTS
```

*Listing 2. ROMable printer driver with 1K buffer.*

the first 64 bytes is the same in both systems, so changing from one Flex to the other involves just changing the size of the FCB.

A second change involves the addresses of Flex routines. MiniFlex is located at addresses 7000-7FFF, while Flex 2.0 is at A080-BFFF. Thus, all references to these locations must be changed. See Table 1 for a fairly complete list of equivalents between the two. Mini-Flex does not keep the date as Flex 2.0 does, so changing from Flex 2.0 back to miniFlex may require deleting all references to dates.

Finally, there is a slight difference in how the two systems handle the SETEXT subroutine. When this routine is executed, miniFlex changes the contents of the index register, while Flex 2.0 doesn't. Thus, if you are converting a program from Flex 2.0 back to miniFlex, you may want to save index register contents before jumping to SETEXT and then restore it.

While converting programs from Flex 2.0 back to miniFlex may not be common, I tried it recently to convert Programma's SPL/M compiler. The conversion was simple.

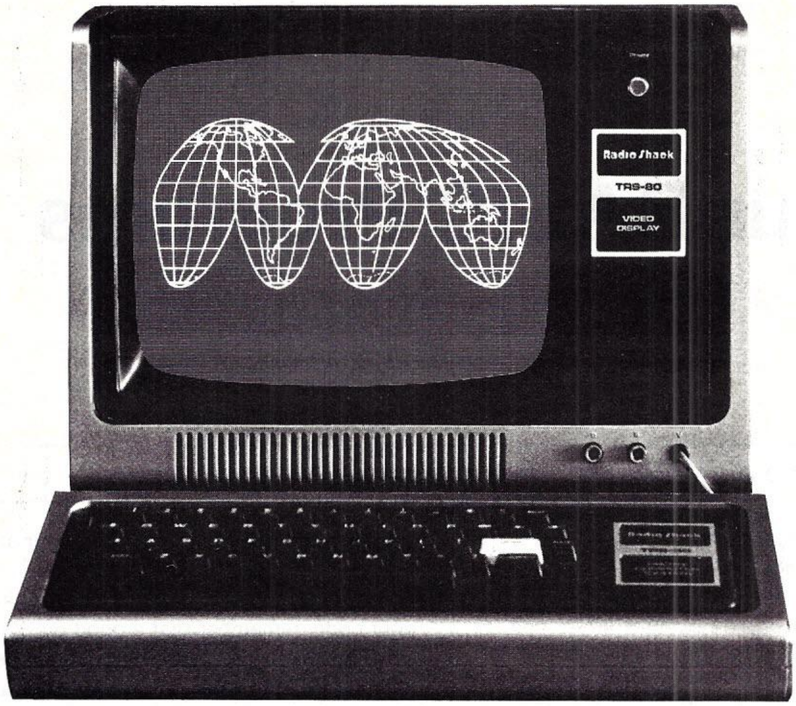
### Next Time

A review and detailed comparison of the SPL/M compiler with several others (Microware's ABASIC, SSB's FORTRAN and Hemenway's STRUBAL +) will appear in upcoming installments. ■

| Name | Flex 1 & 2 Address | MiniFlex Address |
|---|---|---|
| LINBUF | A080 | 7000 |
| FCB | A840 | 7740 |
| EOLCHR | AC02 | 7082 |
| MONTH | AC0E | - |
| DAY | AC0F | - |
| YEAR | AC10 | - |
| LSTERM | AC11 | 7091 |
| LINPTR | AC14 | 7094 |
| CURCHR | AC18 | 709A |
| COLDS | AD00 | 7100 |
| WARMS | AD03 | 7103 |
| RENTER | AD06 | 7106 |
| OUTCH2 | AD12 | 7136 |
| GETCHR | AD15 | 710F |
| PUTCHR | AD18 | 7112 |
| INBUFF | AD1B | 7115 |
| PSTRNG | AD1E | 7118 |
| CLASS | AD21 | 711B |
| PCRLF | AD24 | 711E |
| NXTCH | AD27 | 7121 |
| RSTRIO | AD2A | 7124 |
| GETFIL | AD2D | 7127 |
| LOAD | AD30 | 712A |
| SETEXT | AD33 | 712D |
| ADDBX | AD36 | 7130 |
| OUTDEC | AD39 | 7133 |
| OUTHEX | AD3C | 7139 |
| RPTERR | AD3F | 713C |
| GETHEX | AD42 | 713F |
| OUTADR | AD45 | - |
| INDEC | AD48 | - |
| DOCMND | AD4B | 7142 |
| HEXADJ | B198 | 74BA |
| OPREAD | B2A4 | 758A |
| FMSCLS | B403 | 7803 |
| FMS | B406 | 7806 |

*Table 1. Flex routine addresses.*

```
ED20 7F D05A          CLR   IRQON       TURN OFF CURRENT IRQ OUTPUT FLAG
ED23 39               RTS

                * ACIA IS BUSY, SO STORE IN BUFFER
ED24 37         BUSY  PSH B
ED25 FF D048          STX   SAVEX       SAVE INDEX REGISTER
ED28 FE D051    BUSY1 LDX   INPTR       IS BUFFER FULL?
ED2B 08               INX
ED2C 8C 0400          CPX   #$0400      FIX OVERFLOW OVER 1K
ED2F 26 03            BNE   BUSY2
ED31 CE 0000          LDX   #$0000
ED34 BC D053    BUSY2 CPX   OUTPTR
ED37 26 04            BNE   NOTFUL
                * FULL, SO WAIT UNTIL SOME SPACE IS AVAILABLE
ED39 01               NOP
ED3A 0E               CLI               ENABLE INTERRUPTS
ED3B 20 EB            BRA   BUSY1       GO BACK AND CHECK AGAIN
                * NOT FULL, SO STORE CHARACTER IN BUFFER
ED3D 01         NOTFUL NOP
ED3E 0F               SEI               DISABLE INTERRUPTS IF ENABLED ABOVE
ED3F FE D051          LDX   INPTR
ED42 BD ED97          JSR   FINDCH      GO TO FIND INPTR LOC IN BUFFER
ED45 A7 00            STA A 0,X         STORE INTO NEXT EMPTY BUFFER LOC
ED47 B6 D051          LDA A INPTR
ED4A F6 D052          LDA B INPTR+1
ED4D CB 01            ADD B #1          INCREMENT IN POINTER
ED51 84 03            AND A #$03        MASK TO LIMIT TO 1K
ED53 B7 D051          STA A INPTR
ED56 F7 D052          STA B INPTR+1
ED59 01               NOP
ED5A 0E               CLI               ENABLE INTERRUPTS
ED5B 33               PUL B             RESTORE B AND INDEX
ED5C FE D048          LDX   SAVEX
ED5F 39               RTS               RETURN

                *INTERRUPT SERVICE ROUTINE
ED60 B6 8000    ISS   LDA A ACIACR      DID SOMETHING ELSE INTERRUPT?
ED63 2A 4B            BPL   ERROR       IF YES
                * OK - INTERRUPT CAME FROM ACIA
ED65 7F D050          CLR   BUSYFL      ACIA NO LONGER BUSY.
ED68 FE D051          LDX   INPTR       IS THE BUFFER EMPTY?
ED6B BC D053          CPX   OUTPTR
ED6E 26 06            BNE   NOTEMT      NO
ED70 86 15            LDA A #$15        DISABLE ACIA TRANSMITTER INTERRUPTS
ED72 B7 8000          STA A ACIACR
ED75 3B               RTI               RETURN
                * BUFFER NOT EMPTY, SO OUTPUT THE NEXT CHARACTER
ED76 FE D053    NOTEMT LDX  OUTPTR
ED79 BD ED97          JSR   FINDCH      GO FIND OUTPTR CHAR IN BUFFER
ED7C A6 00            LDA A 0,X         GET NEXT CHARACTER
ED7E B7 8001          STA A ACIADR      OUTPUT IT
ED81 B6 D053          LDA A OUTPTR
ED84 F6 D054          LDA B OUTPTR+1
ED87 CB 01            ADD B #1          INCREMENT OUT POINTER
ED89 89 00            ADC A #0
ED8B 84 03            AND A #$03        MASK TO LIMIT TO 1K
ED8D B7 D053          STA A OUTPTR
ED90 F7 D054          STA B OUTPTR+1
ED93 7C D050          INC   BUSYFL      ACIA IS BUSY AGAIN
ED96 3B               RTI               RETURN TO PROGRAM

                * FINDCH - ADD POINTER TO BUFFER ADDR
ED97 FF D055    FINDCH STX  TEMP        SAVE POINTER
ED9A 36               PSH A             AND SAVE CHARACTER
ED9B B6 D055          LDA A TEMP
ED9E F6 D056          LDA B TEMP+1
EDA1 CB 00            ADD B #BUFFER     ADD BUFFER ADDR TO POINTER
EDA3 89 D4            ADC A #BUFFER/256
EDA5 B7 D055          STA A TEMP
EDA8 F7 D056          STA B TEMP+1      SAVE NEW ADDRESS
EDAB FE D055          LDX   TEMP        PUT INTO INDEX REG
EDAE 32               PUL A             RESTORE CHARACTER
EDAF 39               RTS

                * ERROR ROUTINE - INTERRUPT NOT CAUSED BY ACIA
EDB0 86 15      ERROR LDA A #$15        DISABLE ACIA INTERRUPTS
EDB2 B7 8000          STA A ACIACR
EDB5 FE D051    ERROR1 LDX  INPTR       SEE IF BUFFER IS EMPTY
EDB8 BC D053          CPX   OUTPTR
EDBB 27 1F            BEQ   EMPTY       YES
EDBD FE D053          LDX   OUTPTR      NO, SO EMPTY IT BEFORE QUITTING
EDC0 BD ED97          JSR   FINDCH      FIND OUTPTR CHAR IN BUFFER
EDC3 A6 00            LDA A 0,X         GET NEXT CHARACTER
EDC5 BD FC0C          JSR   OUTEEE      PRINT IT
EDC8 B6 D053          LDA A OUTPTR
EDCB F6 D054          LDA B OUTPTR+1
EDCE CB 01            ADD B #1          INCREMENT OUT POINTER
EDD0 89 00            ADC A #0
EDD2 84 03            AND A #$03        MASK TO LIMIT TO 1K
EDD4 B7 D053          STA A OUTPTR
EDD7 F7 D054          STA B OUTPTR+1
EDDA 20 D9            BRA   ERROR1      GO BACK AND TEST AGAIN
                *BUFFER EMPTY, SO PRINT ERROR MESSAGE AND QUIT
EDDC CE EDE5    EMPTY LDX   #ERRMSG
EDDF BD FC12          JSR   PDATA       PRINT ERROR MESSAGE
EDE2 7E FC03          JMP   WARMST      RETURN TO MONITOR
EDE5 0D         ERRMSG FCB  $D,$A,0,0,0,0
EDEB 45               FCC   /ERROR - IRQ NOT FROM ACIA/
EE04 04               FCB   $04
```


Radio Shack
TRS-80
VIDEO DISPLAY

# Speed Up Your BASIC Programs

*Simple techniques do make a difference; they can work for you, too.*


*Edward H. Carlson*
*3872 Raleigh Dr.*
*Okemos, MI 48864*


| | | |
|---|---|---|
| 1 | FOR A = B TO C | 2.8 |
| 2 | NEXT | 1.1 |
| 3 | NEXT A | 1.5 |
| 4 | REM | 0.4 |
| 5 | DATA 1 | 0.4 |
| 6 | RESTORE | 0.4 |
| 7 | A = A | 1.4 |
| 8 | A = 1 | 2.0 |
| 9 | each additional digit | 1.1 |
| 10 | BB = 1 | 2.1 |
| 11 | each additional character | 0.1 |
| 12 | GO TO 3 | 0.9 |
| 13 | GOSUB... RETURN | 1.2 |
| 14 | PRINT | 72 |
| 15 | PRINT A | 80 |
| 16 | PRINT A$ where A$ = "A" | 72 |
| 17 | each additional character | 0.3 |
| 18 | C = FNA(B) | 3.6 |
| 19 | C = A + B or C = A − B | 2.6 |
| 20 | each additional + A | 1.1 |
| 21 | C = A * B or C = A/B | 3.4 |
| 22 | each additional * A | 1.8 |
| 23 | IF C THEN 5 where C = A<B is false | 1.2 |
| 24 | C = A<B is false | 2.9 |
| 25 | IF A<B THEN 5 false | 2.5 |
| 26 | : for 2 statements on a line | 0.1 saved |
| 27 | space in a line | 0.03 |
| 28 | examine 1 entry in variable table | 0.03 |
| 29 | A = SQR(B) | 39 |
| 30 | A = SIN(B) or A = COS(B) | 22 |
| 31 | A = TAN(B) | 39 |
| 32 | A = ATN(B) | 27 |
| 33 | A = EXP(B) | 22 |
| 34 | A = LOG(B) | 16 |
| 35 | A = B C | 39 |
| 36 | A = B AND C | 4.0 |
| 37 | A = B OR C | 4.0 |
| 38 | C = NOT A | 2.9 |
| 39 | A = USR(B) | 2.3 |
| 40 | A = LEN(B$) | 2.6 |
| 41 | A = ASC(A$) | 2.4 |
| 42 | A$ = LEFT$(B$,1) or MID or RIGHT | 4.5 |
| 43 | A$ = CHR$(B) | 2.8 |
| 44 | A$ = STR$(B) | 7.9 |
| 45 | A = VAL(B$) | 3.3 |

***Table 1. Time (in ms) for BASIC to process each statement or fraction of a statement. Lines 12 and 13 reflect minimum times, since the times depend on how many numbers are examined. SIN functions, for example, may vary with the value of the argument. Line 26 shows the time saved for two lines of code. For all entries, the times may depend on how the statement is used.***

The secrets of speeding up BASIC are simple once you analyze how it goes about its work. Best of all, you may need to change only a small portion of the code to make it run faster.

This article's examples are for the Microsoft BASIC-IN-ROM, Version 1.0, Rev. 3.2, used on my OSI C2-4P and other Ohio Scientific computers. A similar Microsoft BASIC is used on the PET and the TRS-80. In fact, most BASIC interpreters should behave in a similar way. This article describes procedures you can use to check out your own machine.

Typically, a program spends much of its time in a small fraction of its total code. This is especially true when nested loops are present. The running time of the program is sensitive to the efficiency of the innermost loop.

### Speed and Clarity

The BASIC interpreter decodes and executes one line of source code at a time, perhaps even branching out of the line before reaching its end. Within a line, the interpreter reads and decodes each character in turn. The fewer lines and characters the program has, the faster the interpreter can work.

Before you go on a binge condensing your code, remember that a good program has other essential characteristics besides speed. One of the most important is clarity. This is effected in several ways that are the opposite of condensation: for example, long, descriptive variable names, spaces between words, many Remark statements and the use of subroutines to emphasize the logical structure of the program.

So if your program has nested loops, or otherwise spends most of its time in a small portion of the code, then only that code need be condensed. The rest can be written with clarity as the prime consideration (if space in memory is not also a problem).

Besides condensing your code to obtain speed, you must also realize that BASIC uses and stores its numbers in binary form and so needs to decode the decimal form that occurs in the source code. It must do this each time it reinterprets a line; it never remembers that it has decoded the number before. Assign and use names for any such decimal constants, and your programs will be greatly speeded up.

For example, Listing 1 shows three versions of a program to fill my computer's screen with the letter A. Version a runs in 25 seconds, but version b, which avoids repeated decoding of the decimal numbers, runs in ten seconds. Version c is the fastest, running in eight seconds.

It is important to note that the FOR statement is only interpreted once during the loop—at the beginning—while the NEXT statement is executed each time the interior of the loop is traversed to test for completion of the loop. So optimization of the FOR statement is not valuable in speeding up your program.

Other "slowpokes" in BASIC include the statements that reassign the flow of execu-

tion, such as GO TO, GOSUB and ON...GOTO. They can be speeded up, however. They start searching for their target line number at the lowest-numbered line in the source program and spend 0.85 milliseconds per line. This can add up to a large amount of time in a long program.

Instead of starting a program with the one-time-only housekeeping statements, continuing with the central structure and ending with the subroutines, I start my programs with a jump to high line numbers, where I put the initializing and housekeeping statements.

The main body of the program is written next, starting at lower line numbers but still high enough to leave plenty of room at the beginning for all the subroutines (or at least the ones called from time-sensitive inner loops). Another advantage is that the decimal addresses in the GOSUB statements will have fewer digits and will be decoded quicker.

```
10 FOR I=0 TO 2047
20 POKE 53248+I,65
30 NEXT
```

*Listing 1a.*

```
5 Q=53248
7 A=65
10 FOR I=0 TO 2047
20 POKE Q+I,A
30 NEXT
```

*Listing 1b.*

```
5 Q=53248:B=65
10 FORA=QTOQ+2047:POKEA,B:NEXT
```

*Listing 1c.*

**Searching the Variable Table**

When RUN is hit and execution of the program starts, there is no variable table. It is created by the interpreter as it scans lines. Each time a variable name is found, the variable table is searched. If the variable name is not already in the table, it is added at the end of the table.

BASIC always searches the table in order from the first-encountered to the last-encountered variables. Your most-used variables should be placed at the beginning of the table by executing a statement early in the program that defines the desired variables, such as X = 0: Y = 0: Z = 0.

The variable table search is efficient (probably because BASIC knows that each entry is exactly six bytes long). The time spent is only 0.03 milliseconds per entry examined. This is about the same time spent ignoring a space in decoding a line of source code. Paying attention to the variable table will probably not speed up your

programs much.

### Subroutines

It is more effective to place several statements on one line by separating them with a colon (in OSI BASIC). A time savings of 0.1 milliseconds is achieved each time such a colon is encountered. A space savings of three bytes is also achieved.

Using colons, short "subroutines" can be written directly after an IF statement on the same line, rather than using the IF . . . THEN with a GOSUB to jump to a subroutine. This saves time by omitting the search-for line number and also gives a tight execution of the "subroutine." That portion of the statement after THEN is not interpreted if the IF condition was "false."

Listing 2 shows two versions of the program. Version a uses a proper subroutine, while version b differs only in line 20, which has a "subroutine" on the same line with an IF. (Both of these programs are executed with "RUN100" to jump to the housekeeping lines. Version b runs 12 seconds faster than version a, or 1.2 milliseconds saved for each GOSUB . . . RETURN avoided.)

```
5 A=A:RETURN
10 FOR I=0TO10000
20 IF A=1 THEN GOSUB 5
30 NEXT
40 END
100 A=1
110 GOTO 10
```

*Listing 2a.*

```
20 IFA=1THENA=A
```

*Listing 2b.*

### Summary

We have discussed the principles and the most important applications for speeding up BASIC. You can see from Listing 2 how BASIC can be tested. A loop 10,000 turns long is used to execute a statement. Then the statement is replaced by one differing in some feature.

By subtracting the running times, you can find the difference in time to execute the given feature of code. Table 1 gives the running time for several BASIC statements and portions of statements for the C2-4P computer.

Listings 1 and 2 were made on a Comprint 912 printer connected to the 6522 VIA parallel port on the CPU board of the C2-4P computer. ■

# Whoa, Apple

*These two machine-language programs tighten the reins on galloping video displays.*

*Terry Edward Phillips*
*6011-4 Majors Lane*
*Columbia, MD 21045*

Apple II is one of the fastest kids on the block. When you list, DSP or trace a BASIC program, the screen fills at an impressive speed.

But this can sometimes be annoying and frustrating. If you are trying to seriously follow the information that's being output, not even a speed-reading course would help.

Fortunately, two programs can easily give you manual control over your output rate. Both work well with Apple Integer, Applesoft (cassette or firmware) and the Apple Monitor.

**Introduction**

Anything your Apple II displays on its screen resides in a predetermined reserved section of your Apple's memory. Normally, RAM locations 0400-07FF are used for storage. This screen memory area, called the primary page in the Apple manuals, is continuously queried by hardware in your Apple, which then generates the video signals for your monitor.

Interestingly, this process is completely unknown to the microprocessor. The CPU has no idea that someone else is using its memory. The system knows only that to output video data, it must deposit the data into this reserved area.

A machine-language subroutine located in ROM at location FDF0 places alphanumeric characters in the screen memory area one at a time. This and subsequently-called subroutines are responsible for placing the byte in the accumulator (passed by the calling routine) into the screen memory, adjusting the line width, line feeds and scrolling and doing other housekeeping chores.

The BASIC interpreter and Apple monitor routines call this subroutine every time they want to communicate with you via the screen. For example, if your BASIC interpreter is executing a PRINT"APPLE" statement, the interpreter will place the ASCII code for the letter A in the accumulator and go to the subroutine at FDF0. When it returns from this subroutine (the letter A is now on the screen), the code for the letter P is then placed in the accumulator, and so forth.

The BASIC interpreter and Apple monitor do not go directly to the location FDF0 to output a character. Rather, they look at RAM locations 0036, 0037 to find their output subroutine. When the reset key is pushed, the Apple monitor will, among other tasks, place the address of the normal video output routine, FDF0, into locations 0036, 0037. The calling routine then finds its output subroutine by executing an indirect jump to location 0036.

This means it jumps, not to location 0036, but to the 16-bit address contained in locations 0036, 0037. Thus, the calling routine, in a roundabout way, finds the normal video output routine.

At first, this technique appears to be unnecessarily complex. Why not have the in-

```
Address  Code      Label   Mnemonic    Comment

  2DC    A9 E5             LDA#E5      Place Delay Subroutine
  2DE    85 36             STA 36      Address in locations
  2E0    A9 02             LDA#02      0036, 0037
  2E2    85 37             STA 37
  2E4    60                RTS

  2E5    08        ENTRY   PHP         Save relevant registers
  2E6    48                PHA
  2E7    18        FGCLR   CLC         Clear Flag
  2E8    AD 00 C0  RKYBD   LDA C000    Read Keyboard
  2EB    10 0C             BPL FLGCK   Branch to Check Flag if
                                       no key is depressed
  2ED    2C 10 C0          BIT C010    Clear Keyboard Strobe
  2F0    49 91             EOR#91      Mask for control keys
  2F2    F0 F3             BEQ FGCLR   Branch to Clear Flag if
                                       CTRL Q was depressed
  2F4    29 FD             AND#FD      Mask for CTRL S key
  2F6    D0 01             BNE FLGCK   Branch to Check Flag if
                                       not CTRL S
  2F8    38                SEC         Set Flag if CTRL S is hit
  2F9    B0 ED     FLGCK   BCS RKYBD   Branch to Read Keyboard
                                       if Flag is Set
  2FB    68                PLA         Restore relevant registers
  2FC    28                PLP
  2FD    4C F0 FD  EXIT    JMP FDF0    Jump to Normal Video
                                       Output Subroutine
```

*Program 1. Halts video output when the CTRL S key is depressed. Output is resumed when the CTRL Q key is hit.*

terpreter and monitor go directly to FDF0? But this sleight of hand is useful. With this indirect method, you can route output data to any device, such as a printer, paper punch or TTY, by placing the vector address of the output routine designed for that particular device in locations 0036, 0037.

This technique of vectoring the output offers a solution to the problem at hand. You can change the vector address in locations 0036, 0037 from FDF0 to the address of a short routine whose only purpose is to provide some sort of delaying tactic. After the delay condition is satisfied, the character in the accumulator is sent to the normal video output subroutine located at FDF0. The net effect is to reduce the speed of video output.

**Two Patches**

Two short machine-language patches are designed to provide a delay and give you manual control over the speed of video output.

The first program checks the keyboard when a character is to be output to see if the CTRL S key was depressed. If not, the normal output is continued. If the key was depressed, the routine waits in a continuous loop until you depress the CTRL Q key. Video output, therefore, can be stopped and started by alternately striking the CTRL S and CTRL Q keys respectively.

The second program is a bit different. It uses game paddle 0 to determine the output speed. When a character is to be output, this program reads the value of paddle 0 via a subroutine in ROM. This becomes the initial value of a two-stage timing loop. When the counters in the loop are decremented to zero, the loop is exited and the character is passed to the normal output routine. This variable timing loop allows you to vary the output speed from approximately three characters per second to nearly full speed, just by varying the position of game paddle 0.

**Memory Location**

Either routine, as presented, is assembled at the top of page two of memory (0200–02FF). This area of memory is normally used by the Apple monitor as a keyboard buffer. All your keystrokes between successive returns are stored here. As long as you don't type more than about 200 keys before hitting return, page two of memory can be shared amicably between the two functions. I decided to place the routines there to avoid possible conflicts with other software packages, but if its location is inconvenient or undesirable, you can relocate the programs elsewhere with only minor changes.

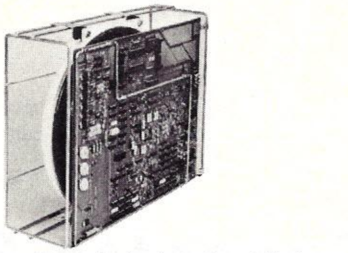
The selected routine can initially be entered into memory with the ROM-based assembler described in the Apple manual. Subsequent loads can be done via the tape cassette, although they are short enough that manual loading might be faster.

The selected routine is then initialized by executing a call to 02DC in the monitor mode or a CALL 732 in BASIC. You resume normal video output by executing a ROM-based subroutine with a call to FE93 or a CALL −365 in BASIC or when reset is hit. ■

```
Address  Code      Label   Mnemonic     Comment

2DC      A9 E5             LDA#E5       Place Delay Subroutine
2DE      85 36             STA 36       Address in locations
2E0      A9 02             LDA#02       0036, 0037
2E2      85 37             STA 37
2E4      60                RTS

2E5      48        ENTRY   PHA          Save registers
2E6      98                TYA
2E7      48                PHA
2E8      8A                TXA
2E9      48                PHA
2EA      A2 00             LDX#00       Set index for Paddle 0
2EC      20 1E FB          JSR FB1E     Subroutine to read Paddle
2EF      C8                INY          Adjust value of Paddle
2F0      98                TYA          Copy value into A
2F1      AA        RSTRX   TAX          Restore value into X
2F2      CA        INLOP   DEX          Inner delay loop
2F3      D0 FD             BNE INLOP
2F5      88                DEY          Outer loop decrement
2F6      D0 F9             BNE RSTRX    Restore inner loop if
                                        outer loop is incomplete
2F8      68                PLA          Restore registers
2F9      AA                TAX
2FA      68                PLA
2FB      A8                TAY
2FC      68                PLA
2FD      4C F0 FD  EXIT    JMP FDF0     Jump to Normal Video
                                        Output Subroutine
```

*Program 2. Utilizes the game paddle 0 to vary the video output speed from approximately three characters per second to full speed.*

# PRIAM Hard Disks Now Available from SIRIUS SYSTEMS!



PRIAM's high-performance, low-cost Winchester disc drives speed up throughput and expand data storage from 20 megabytes to 154 megabytes. And a single controller can be used to operate 14-inch-disc drives with capacities of 33, 66, or 154 megabytes or floppy-disc-size drives holding 20 and 34 megabytes. So it's easy to move up in capacity, or reduce package size, without changing important system elements or performance.

- *Fast, Linear Voice Coil Positioning*
- *10 ms track-to-track positioning*
- *Fully servoed head positioning*
- *Dedicated servo tracks*
- *DC Power required only!*
- *Simple, parallel Interface*
- *Optional SMD Interface*
- *50 ms Average Positioning time*
- *90 ms Maximum Positioning Time*
- *6.4 ms Average Latency*

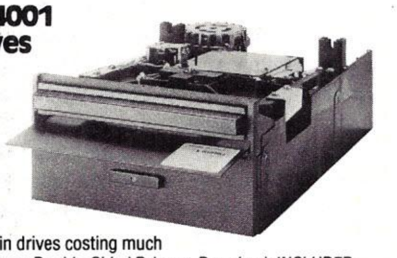
## THE PRIAM LINEUP

| Model/Disc Size | Capacity | Size | Weight | Price |
|---|---|---|---|---|
| DISKOS 3350 (14") | 33Mbytes | 7" × 17" × 20" | 33 lbs. | $2995 |
| DISKOS 6650 (14") | 66 Mbytes | 7" × 17" × 20" | 33 lbs. | $3749 |
| DISKOS 15450 (14") | 154 Mbytes | 7" × 17" × 20" | 33 lbs. | $4695 |
| DISKOS 2050 ( 8") | 20 Mbytes | 4.62" × 8.55" × 14.25" | 20 lbs. | $2995 |
| DISKOS 3450 ( 8") | 34 Mbytes | 4.62" × 8.55" × 14.25" | 20 lbs. | $3745 |
| DISKOS 570 | 5.3 Mbytes | floppy-size | (low) | (low) |
| DISKOS 1070 | 10.6 Mbytes | floppy-size | (low) | (low) |

All PRIAM DISKOS Drives have a Transfer Rate of 1.03 Mbytes/Sec.
*Optional SMD interface available for $150.*

SIRIUS SYTEMS offer cases and enclosures for all PRIAM Hard Disk Drives. All 14" Winchester Drives will mount in our 14" Standard Case. The 8" Winchesters have two alternatives: a single drive case and a dual drive case. All SIRIUS SYSTEMS Winchester drive cases include Power Supply, internal cabling, switches, fan, extra AC outlet (not switched, but fused) and possess very adequate ventilation. Drive addressing is done on the rear of the Case and not on the drive iteself to provide ease of use during operation. All WINCHESTER DRIVE Cases are Warranted for a full year and come in our standard blue-black color scheme. Consult us for current availability and pricing.

## Remex RFD 4000/4001 8" Floppy Disc Drives Double sided... Double density!!



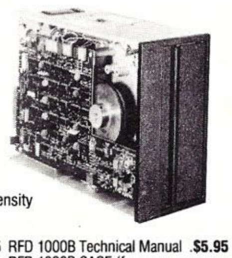
**$549.95**

**RFD 4001, $569.95**

Offers quality and features found in drives costing much more! ■ Single or Double Density ■ Double-Sided Drive ■ Door Lock INCLUDED ■ Write-Protect INCLUDED ■ 180 Day Warranty ■ Compatible with Shugart 850/851 ■ Low Power Operation ensures LONGER LIFE!! ■ Model RFD 4001 offers Data and Sector Separator

RFD 4000/4001 Technical Manual ........ **6.95**
Connector Set #3 (AC, DC, Card Edge) .. **10.95**
Connector Set #4 (AC and DC) .......... **2.95**
RFD 4000C/B Cabinet (for use with Power Modules) .................... **29.95**

## Remex 1000B... If you've been looking for a less expensive floppy disc drive, but not wanting to sacrifice quality — this is it!



**$419.95**

You get both in the Remex 1000B! For only $419.95 look at what you get: ■ 8" Floppy Drive ■ Single or Double Density ■ Hard or Soft Sectoring ■ Media Protection Feature ■ Single Density Data Separator ■ 180 Day Factory Warranty

Door Lock Option ..... **$19.95**
Interface Adapter (REMEX-to-Shugart) .. **$14.95**
Write Protect Option .... **$19.95**
Connector Set #1 (AC, DC, & Card Edge) .. **$10.95**
RFD 1000B Technical Manual **.$5.95**
RFD 1000B CASE (for use (with Power Modules) ..... **$29.95**

## SIRIUS 8" DISK POWER MODULES

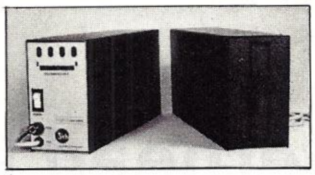
The Single and Dual Drive Power Modules are designed to provide DC and (switched) AC power for one (the Single Drive Power Module) or two (the Dual Drive Power Module— the DDPM will power three RFD 4000s or 4001s) 8" Floppy Disk Drives. Many features are included for safe and reliable operation and the Power Modules come with our standard 180 day WARRANTY (the Open Frame Power Supply warranty is for 2 years). All Power Modules will work with either the RFD 4000C/B or RFD 1000B case (color schemes match also).

Dual Drive Power Module (DDPM) ........ **$139.95**

Single Drive Power Module (SDPM) ......... **119.95**

# SIRIUS 80+

## Perfect Add-Ons for Your Computer System!



The SIRIUS SYSTEMS 80+ Series of Floppy Disk add-ons are designed to provide unmatched versatility and performance for your computer. Consisting of four different add-ons, there is a 80+ Series Floppy Disk to meet your need. All 80+ Series Floppy Disk are compatible with the TRS-80* and come ready to plug in!

**COMMON CHARACTERISTICS**

- *5 ms track-to-track access time*
- *Auto-eject*
- *180 day WARRANTY*
- *Exceptional speed stability — 1½%*
- *Single density (FM) or double density (MFM/M²FM)*
- *Ultra high reliability*
- *2 year Power Supply Warranty*
- *Mix any or all 80+ Series on the same cable!*
- *Includes user accessible plugboard for drive reconfiguring*

**SPECIFIC CHARACTERISTICS**

The **SIRIUS 80+1** is a single sided, 40 track, highly reliable Floppy Disk add-on. Offering 5 more tracks than the Radio Shack model, it cost $140 less! Formatted data storage is 102K/20K bytes single/double density.

SIRIUS 80+1 ............... **$359.95**

The **SIRIUS 80+2** is a dual sided, 70 track (35 per side), highly versatile Floppy Disk unit. It appears to the TRS-80* as TWO 35 track drives, yet COST LESS THAN HALF THE PRICE! Even greater savings result, since data is recorded on both sides of the media instead of only a single side. Using the plug board, it may be reconfigured for other computer systems! (The 80+2 operates as Drive 0 and any of the other three addresses (with the sandard Radio Shack Cable) or as any of four drives (with the SS Standad Cable).) Formatted data storage is 80.6K/161.2K bytes single/double density.

SIRIUS 80+2 ............... **$449.95**

The **SIRIUS 80+3** is a single sided, 80 track, "Quad" density Floppy Disk unit. Offering 2⅓ times the storage of a Standard Radio Shack drive, the 80+3 greatly reduces the need for diskettes correspondingly. Additionally, because of the increased storage and faster track-to-track access time, the 80+3 allows tremendously increased throughput for disk based programs!!! The 80+3 INCLUDES SIRIUS's TRAKS-PATCH on Diskette. Formatted data storage is 204K/40K8 bytes single/double density.

SIRIUS 80+3 ............... **$489.95**

The **SIRIUS 80+4** Floppy Disk add-on is a double sided, 160 track (80 per side), 5¼" monster! The ultimate in state-of-the-art 5¼" Floppy Disk technology, to 80+4 is seen by the TRS-80* as two single sided disk drives, each with 80 tracks. Thus, in terms of capacity one 80+4 is equivalent to 4⅔ standard Radio Shack drives — a savings of over 73% (not to mention diskettes!!!). (With a double density converter, the available memory is huge!) The 80+4 is similar to the 80+2 in that it arrives configured as Drive 0 and any of the other three addresses (with the standard Radio Shack Cable) or as any of four drives (with the SS Standard Cable). The 80+4 INCLUDES TRAKS-PATCH on Diskette. (The plug board is also included.) Formatted data storage is 408K single density or 816K bytes double density.

SIRIUS 80+4 ............... **$624.95**

Ø

All 80+ Series Floppy Disk add-ons operate a 5 milliseconds track-to-track access time (eight times faster than the SA 400) but are Expansion Interface Limited to 12 milli-seconds for the TRS-80*.

*TRS-80© Tandy Corp.

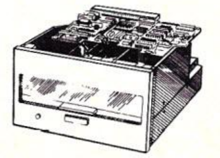
# MPI 51/52...

## A Great Reliable Mini-Drive!

- *Fast! 5ms track to track access*
- *Exclusive Pulley-Band Design*
- *Unique Door/Ejector Mechanism*
- *Reliable 1½% Speed Stability*
- *Single/Double Density Operation*
- *Industry/ANSI Standard Interface*

**MPI 51**
(Single Head, 40 tracks, 120K/240K bytes Single/Double Density**) ..... **$259.95**

**MPI 52**
(Dual Head, 70 tracks, (35/side), 218.8K/437.5K Single/Double Density**) ...................... **$349.95**



# MPI 91/92... NEW STATE-OF-THE-ART DISK DRIVE!

**MPI 91**
(Single Head, 80 tracks, 240K/480K Single/Double Density**) ......... **$389.95**

**MPI 92**
(Single Head, 160 tracks (80/side), 480K/960K Single/Double Density**) ...................... **$499.95**

***Unformatted data storage*

# Introducing the Versatile, Low-Cost OMEGA Series Controller

As new technological advances bring down the cost of fast, reliable mass data storage, the need for an inexpensive, versatile controller have become greater and greater. To meet this need, SIRIUS SYSTEMS' OMEGA Series Controller was designed.

The SIRIUS OMEGA Series Controller Module utilizes an **on-board microprocessor** to mediate data transfer to a wide variety of peripherals from an equally wide variety of host computer systems. Up to four Winchester Hard Disks (8" or 14"), four 5¼" Floppy Disks Drives and/or up to eight 8" Floppy Disk Drives may be **in use at one time.** Host systems interfacing is accomplished via a parallel or a serial interface. With the additon of a Personality module, the OMEGA Series Controller Module is directly **compatible with many popular computer systems** (among them the TRS-80*, Apple, Heath, and others). Provision is made for the addition of a streaming tape drive, also.

**SPECIFIC HARDWARE FEATURES INCLUDE:**

- *Control of up to twelve Floppy Disk Drives (eight 8" and/or four 5¼")*
  - *8" and/or 5¼" Disk Drive Utilization*
  - *Single (FM) or Double (MFM) density data storage*
  - *Hard or Soft sectored diskette usage*
  - *Utilization of "Quad" density (96 tpi) 8" or 5¼" Disk Drives*
- *Control of up to four WINCHESTER type PRIAM DISKOS Disk Drives*
  - *8" or 14" may intermix on the same cable*
  - *Accommodates 8" and/or 14" drives of 5.3Mbytes to 154Mbytes*
  - *Ultra-Fast data transfers*
- *Extremely flexible host-controller interfacing*

**SPECIFIC SOFTWARE FEATURES INCLUDE:**

- Dynamic format modifications via command words
- Extremely flexible format acceptance for unusual data storage formats
- Easily interfaces to standard operating systems (TRS-DOS*, CP/M***, etc)
- Operates in either get/put sector mode or data string mode
- Performance parameters may be changed by EPROM replacement or Dynaminic Reprogramming

Dedicated systems cards are also available on a limited basis for the STD-BUS and the S 100. These cards feature shared memory also (again, software selectable) in addition to the regular OMEGA Series Controller Module features. Consult SIRIUS SYTEMS for current price and availability for the entire line of OMEGA Series Memory Units and Controllers. **Dealer inquiries are invited.**

## TO ORDER CALL (615) 693-6583

Phone Orders Accepted 9AM-7PM (ESDT)

We accept **MC, VISA, AE, COD** (requires Certified Check, Cashier's Check or Cash) and **Checks** (personal checks require 14 days to clear). **SHIPPING AND HANDLING:** $7.00 per Floppy Disk Drive or 80+ Module ■ 5% for other items (any excess will be refunded) ■ **Foreign Orders** add 10% for Shipping & Handling. Payment in U.S. currency ■ Tennessee residents add 6% Sales Tax ■ **VOLUME DISCOUNTS AVAILABLE**

SIRIUS SYSTEMS
7528 Oak Ridge Highway
Knoxville, Tennessee 37921 ✓67

# Low Overhead Cassette Format For 6800 Systems

*No longer do you have to suffer the inconvenience of using 300 baud Kansas City Standard.*

*Dr. Gordon W. Wolfe*
*Department of Physics*
*The University of Mississippi*
*University, MS 38677*

Owners of cassette-based 6800 microcomputers such as the SWTP 6800 have for years put up with the slow loading and punching process known as the Kansas City Standard and the MIKBUG cassette format. The Kansas City Standard is akin to a frequency-shift encoding, but uses an eight-fold redundancy to assure proper character recognition. This eight-fold redundancy results in character rates of only 300 baud.

But the MIKBUG format is the real culprit in most 6800-based systems. MIKBUG was one of the first, and certainly the most successful, of the 6800 operating systems, and manufacturers of other operating systems these days take pains to remain MIKBUG-compatible. Such operating systems as SWTBUG, SMARTBUG and RT-68/MX all have the same I/O routines in the same locations, and all use the same tape punch format.

This has virtually locked the 6800 user into this format (Fig. 1). To output 16 bytes of data to tape, there is a "start of record"—the characters S1. Next comes the count of the number of bytes to follow. And here is where MIKBUG slows down.

Nineteen (decimal) bytes will follow, so the computer punches that 13 (hex) bytes follow—but it punches *a one and a three in ASCII.* That is, for this and all succeeding bytes, it punches *two* characters. Following the count, there is the start address of the record (two bytes; four characters), and then 16 bytes of data (32 characters) and a checksum (two characters). Also, at the end there is a carriage return and line feed.

Total it up: to punch, or record on cassette, only 16 bytes, MIKBUG must output a total of 44 characters, or 2.75 characters for every byte of data. This is very wasteful overhead. If this overhead could be removed, cassette loading time (or TTY loading from paper tape) could be speeded up by a factor of 2.75.

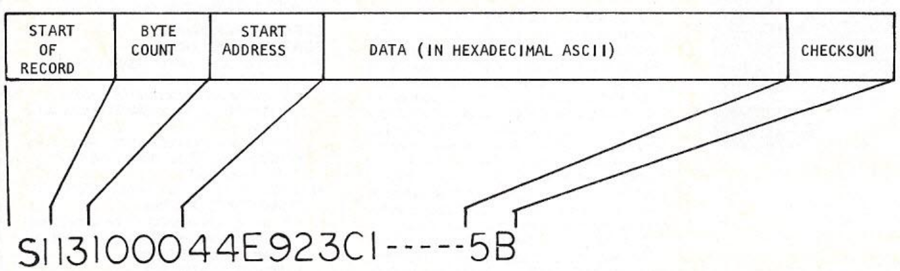



*Fig. 1. MIKBUG punch format.*

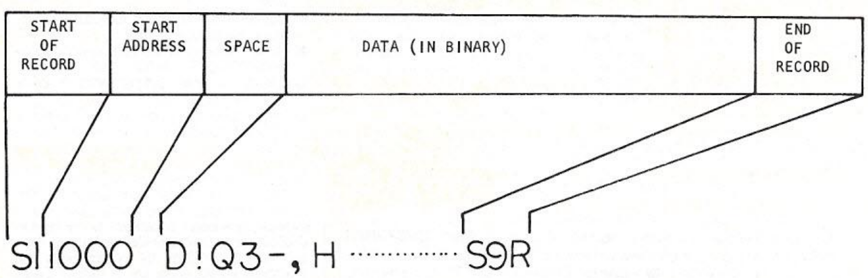



*Fig. 2. BIPNCH punch format.*

Some suggested remedies, such as increasing the record length from 16 to 256 characters, or outputting a pseudo-binary (with the parity bit, bit 7, set to zero) record, have not completely succeeded. The best solution is to simply output straight binary data with as little overhead as possible.

Fig. 2 shows a scheme this author has used successfully. As in MIKBUG, an S1 is used as a start-of-record. Anything that occurs on the tape before this S1 is ignored. Then follows the start address for the record (four ASCII characters representing two bytes) and a space. Then follows, byte for byte, one character for one byte in the same order as in memory. Any number of bytes are punched—one byte, up to all of memory. When data is finished, an end-of record "S9R" is punched.

**The Program BIPNCH**

An assembly-language program named BIPNCH has been written to take care of

| | | |
|---|---|---|
| BPUNCH | $7F00 | Punch subroutine |
| DONE9 | $7F1D | Subroutine prints end of record |
| SECDEL | $7F51 | Subroutine delays one second |
| TLOOP | $7F53 | Subroutine delays a number of half-seconds equal to ACCB |
| BILOAD | $7F62 | Load subroutine |
| | $7F9E | Reset MP-C PIA FOR ECHO |

*Table 1.*

```
(MIKBUG Monitor)

* M A048
A048 DF 7F
A049 13 2A
A04A 22
*G
!0100 01FF   (punches $0100-$01FF)
*
```

*Sample Run 1.*

data loading and punching in this binary format, and is shown in Program 1. The program was written for use on an SWTP 6800 micro using RT68/MX as an operating system. It presumes memory at $A000, and that an MP-C control interface at $8004 is used for communication to the terminal. However, as will be shown later, the program may be easily modified to other systems.

BIPNCH is an example of structured programming; that is, it is written as two monitor-called routines, and a series of subroutines.

The punch routine, BOUT, is called from the monitor by executing location $7F2A. The program outputs a prompt (!) and waits for the user to input the start and end addresses of the region to be punched. When this is done, the program calls the punch subroutine to output headers, data and end-of-record, and returns to the monitor. Sample run 1 shows how it works.

The loading routine, BIN, is called from the monitor by executing location $7FAB. The program disables the MP-C echo and loads the data automatically, returning to the monitor when done.

Both routines will, of course, automatically start and stop the recorder if used with an AC-30 interface.

It should be noted that the punch routine, if used with a terminal and AC-30 interface, will cause characters to print on the screen. Control characters will cause controlled devices to actuate (including recorder on/off), so it is recommended that the control functions be deactuated when punching. On the CT-1024 terminal by SWTP, this may be done by adding a switch that disconnects pin 3 of IC8 on the CT-CA cursor control board. There is no difficulty during the read routine, since the echo is disabled.

In addition to the two monitor call routines listed above, there are user-accessible subroutines which may be of value in other programs. They include the punch and load subroutines themselves. In MIKBUG, the user must go to the monitor to write to or read from tape, or write his or her own routines. Here, the routines are usable directly. To punch, load the start and end addresses to punch into $A016 and $A014, respectively, and jump to the subroutine at $7F00. To load, or read a tape, jump to the subroutine at $7F62. Table 1 is a list of subroutines, their locations and what they do. Note that a monitor-callable program

```
                      NAM  BIPNCH
                *
E07E            PDATA1 EQU  $E07E
E0CC            OUTS   EQU  $E0CC
E0C8            OUT4HS EQU  $E0C8
E047            BADDR  EQU  $E047
E1D1            OUTEEE EQU  $E1D1
A016            STADR  EQU  $A016
A014            ENADR  EQU  $A014
E0E3            CONTL  EQU  $E0E3
                *INPUT ROUTINE DOES
                *NOT STRIP PARITY BIT
E359            INEEE  EQU  $E359
                *
7F00                   ORG  $7F00
                *
                *PUNCH SUBROUTINE
7F00 CE 7F B0   BPUNCH LDX  #OPHDR
7F03 BD E0 7E          JSR  PDATA1
7F06 CE A0 16          LDX  #STADR
7F09 BD E0 C8          JSR  OUT4HS
7F0C FE A0 16          LDX  STADR
7F0F A6 00      MORE1  LDAA 0,X
7F11 BD E1 D1          JSR  OUTEEE
7F14 BC A0 14          CPX  ENADR
7F17 27 03             BEQ  DONE1
7F19 08                INX
7F1A 20 F3             BRA  MORE1
7F1C 39         DONE1  RTS
                *
                *END OF RECORD
7F1D CE 7F B3   DONE9  LDX  #ESS9
7F20 BD E0 7E          JSR  PDATA1
7F23 8D 2C             BSR  SECDEL
7F25 86 14             LDAA #$14
7F27 7E E1 D1          JMP  OUTEEE
                *
                *MONITOR CALL FOR
                *PUNCH ROUTINE
7F2A CE 7F B7   BOUT   LDX  #PROMPT
7F2D BD E0 7E          JSR  PDATA1
7F30 BD E0 47          JSR  BADDR
7F33 FF A0 16          STX  STADR
7F36 BD E0 CC          JSR  OUTS
7F39 BD E0 47          JSR  BADDR
7F3C FF A0 14          STX  ENADR
7F3F 8D 10             BSR  SECDEL
7F41 86 12             LDAA #$12
7F43 BD E1 D1          JSR  OUTEEE
7F46 8D 09             BSR  SECDEL
7F48 8D B6             BSR  BPUNCH
7F4A 8D D1             BSR  DONE9
7F4C 8D 03             BSR  SECDEL
7F4E 7E E0 E3          JMP  CONTL
                *
                *SUBROUTINE DELAYS
                *ONE SECOND
7F51 C6 02      SECDEL LDAB #2
                *HALF SECOND DELAY
                *AT TLOOP IF B=1
7F53 CE F4 FF   TLOOP  LDX  #$F4FF
7F56 09         DECX   DEX
7F57 26 FD             BNE  DECX
7F59 5A                DECB
7F5A 26 F7             BNE  TLOOP
7F5C 39                RTS
                *
                *MONITOR CALL TO
                *OUTPUT "S9R" DATA END
7F5D 8D BE      S9     BSR  DONE9
7F5F 7E E0 E3          JMP  CONTL
                *
                *SUBROUTINE FOR
                *BINARY DATA IN
7F62 CE 7F BB   BILOAD LDX  #RDON
                *SET PIA NO ECHO
7F65 C6 3C             LDAB #$3C
7F67 F7 80 07          STAB $8007
7F6A BD E0 7E          JSR  PDATA1
                *INPUT DATA HEADER
7F6D BD E3 59   AGAIN  JSR  INEEE
7F70 81 53             CMPA #'S
7F72 26 F9             BNE  AGAIN
7F74 BD E3 59          JSR  INEEE
7F77 81 31             CMPA #'1
7F79 26 F2             BNE  AGAIN
7F7B BD E0 47          JSR  BADDR
7F7E BD E3 59          JSR  INEEE
7F81 81 20             CMPA #$20
7F83 26 E8             BNE  AGAIN
                *DATA INPUT
7F85 8D 1D      MORE2  BSR  INPUT
7F87 81 53             CMPA #'S
7F89 26 FA             BNE  MORE2
7F8B 8D 17             BSR  INPUT
7F8D 81 39             CMPA #'9
7F8F 26 F4             BNE  MORE2
7F91 8D 11             BSR  INPUT
7F93 81 52             CMPA #'R
7F95 26 EE             BNE  MORE2
7F97 8D B8             BSR  SECDEL
7F99 86 13             LDAA #$13
7F9B BD E1 D1          JSR  OUTEEE
                *PIA RESET
7F9E C6 34             LDAB #$34
7FA0 F7 80 07          STAB $8007
7FA3 39                RTS
7FA4 BD E3 59   INPUT  JSR  INEEE
7FA7 A7 00             STAA 0,X
7FA9 08                INX
7FAA 39                RTS
                *
                *MONITOR CALL FOR
                *BINARY DATA INPUT
7FAB 8D B5      BIN    BSR  BILOAD
7FAD 7E E0 E3          JMP  CONTL
                *
7FB0 53         OPHDR  FCC  'S1'
7FB1 31
7FB2 04                FCB  4
7FB3 53         ESS9   FCC  'S9R'
7FB4 39 52
7FB6 04                FCB  4
7FB7 0D         PROMPT FCB  $D,$A,$21,$4
7FB8 0A 21
7FBA 04
7FBB 11         RDON   FCB  $11,$4
7FBC 04
                *
                       END
         NO ERROR(S) DETECTED


SYMBOL TABLE:
AGAIN   7F6D     BADDR   E047
BILOAD  7F62     BIN     7FAB
BOUT    7F2A     BPUNCH  7F00
CONTL   E0E3     DECX    7F56
DONE1   7F1C     DONE9   7F1D
ENADR   A014     ESS9    7FB3
INEEE   E359     INPUT   7FA4
MORE1   7F0F     MORE2   7F85
OPHDR   7FB0     OUT4HS  E0C8
OUTEEE  E1D1     OUTS    E0CC
PDATA1  E07E     PROMPT  7FB7
RDON    7FBB     S9      7F5D
SECDEL  7F51     STADR   A016
TLOOP   7F53
```

*Program 1.*

| | | |
|---|---|---|
| PDATA 1 | $E07E | Output data string terminated by EOT |
| OUTS | $E0CC | Output a space |
| OUT4HS | $E0C8 | Output 2 bytes in ASCII hex, with a space |
| BADDR | $E047 | Input 2 bytes in ASCII hex |
| OUTEEE | $E1D1 | Output a character from ACCA |
| CONTL | $E0E3 | Jump to monitor |

*Table 2.*

places "S9R" at the end of a tape; this is useful even with MIKBUG.

### How Does It Work?

In a word, magnificently. Instead of 22 minutes to load 8K BASIC, it now requires only eight minutes. It is possible to double the amount of work I get out of my machine.

The program fits very nicely into a 256 × 8 1702A EPROM at $7F00, with considerable room left over for other routines. Only four bytes of RAM are used at $A014–$A017, and the code is not self-modifying, so that it may be placed in EPROM.

### Modifications to Other Systems

As stated above, the program may be easily modified for other 6800 systems.

If you desire to place the program into a location other the $7F00, you may reassemble it to any other location, or modify the addresses of the instructions at $7F00, $7F1D, $7F2A and $7F62.

Virtually all the external routines in BIPNCH are MIKBUG-compatible. Table 2 lists these routines, and what they do. There should be no trouble with MIKBUG, SWTBUG, SMARTBUG or RT-68/MX.

If you are using an ACIA, rather than an MP-C control interface, the PIA echo/no echo routines at $7F65 and $7F9E must be rewritten.

The only critical external routine which will require extensive modification is the character input routine. It was discovered that both MIKBUG and RT-68/MX strip the most significant bit (set bit 7 equal to zero) whenever a character is input via the subroutine INEEE at $E1AC. Naturally, this is unacceptable, because when you save programs, approximately half the instruction bytes will have bit 7 equal to 1.

In RT-68/MX, the solution is simple. Enter the subroutine at $E359, after the parity bit stripping function. Then the character is input exactly as recorded. For MIKBUG and SMARTBUG, however, the solution is not so simple. It is not possible to bypass the parity stripping so simply. I have no information on SWTBUG, but I presume it is similar.

For those who use these other monitors, Program 2 shows an input routine for MP-C interface which does not strip the parity bit. It is ORGed at $E359, but is relocatable to any other location. I must emphasize that this is similar to, but not the same way as it is done in RT-68/MX, to avoid copyright violations.

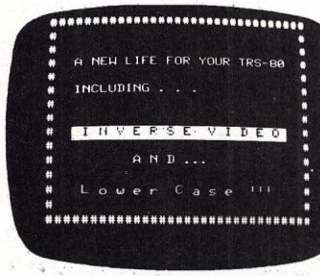
```
                         NAM  INEEE2
                  *
                  *MP-C INPUT ROUTINE
                  *DOESN'T STRIP PARITY
                  *RELOCATABLE
                  *NON-REENTRANT
                  *INPUTS CHAR TO ACCA
                  *FROM MP-C AT $8004
                  *
A010              XTEMP  EQU  $A010
E359                     ORG  $E359
                  *
E359 37                  PSHB
E35A FF A0 10            STX  XTEMP
E35D CE 80 04            LDX  #$8004
                  *
                  *WAIT FOR START BIT
                  *AND START TIMER
E360 A6 00        PIA    LDAA 0,X
E362 2B FC               BMI  PIA
E364 6F 02               CLR  2,X
E366 8D 22               BSR  BIT
E368 8D 1C               BSR  WAIT
E36A C6 04               LDAB #4
E36C E7 02               STAB 2,X
                  *
                  *INPUT 8 BITS
E36E 58                  ASLB
E36F 8D 15        PIA2   BSR  WAIT
E371 0D                  SEC
E372 69 00               ROL  0,X
E374 46                  RORA
E375 5A                  DECB
E376 26 F7               BNE  PIA2
E378 8D 0C               BSR  WAIT
                  *
                  *TEST # STOP BITS
E37A E6 02        CHECK  LDAB 2,X
E37C 58                  ASLB
E37D 2A 02               BPL  RES
E37F 8D 05               BSR  WAIT
                  *
                  *RETURN
E381 FE A0 10     RES    LDX  XTEMP
E384 33                  PULB
E385 39                  RTS
                  *
                  *TIMERS
E386 6D 02        WAIT   TST  2,X
E388 2A FC               BPL  WAIT
E38A 6C 02        BIT    INC  2,X
E38C 6A 02               DEC  2,X
E38E 39                  RTS
                  *
                         END
        NO ERROR(S) DETECTED


   SYMBOL TABLE:
BIT     E38A          CHECK   E37A
PIA     E360          PIA2    E36F
RES     E381          WAIT    E386
XTEMP   A010
```

*Program 2.*

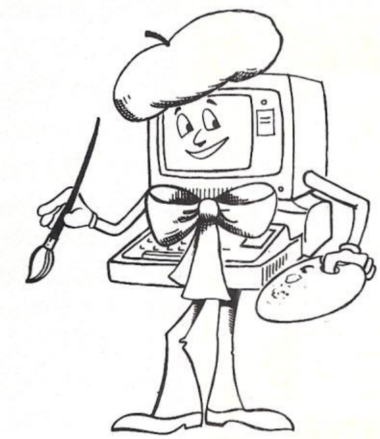
Last, there are some difficulties with using the binary loading technique. I have pointed out the control character problem above. Just as serious is the lack of checksum. The program has no way of knowing if there were any loading errors.

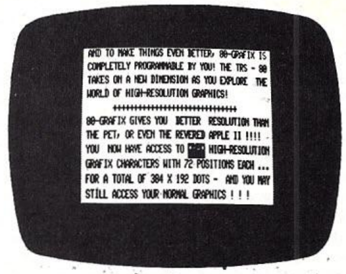
I must say, though, that in eight months of using the program, I have never had a bad load. Even if I did, it only takes a few minutes to re-load. ■

# FROM PROGRAMMA
## HI-RESOLUTION GRAPHICS FOR THE TRS-80®

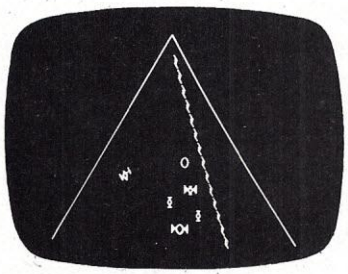



**LOWER CASE**
The 80-GRAFIX board includes two sets of lower case characters at no additional cost.






**INVERSE VIDEO**
The 80-GRAFIX board allows you to do inverse video to high-light your screen displays.



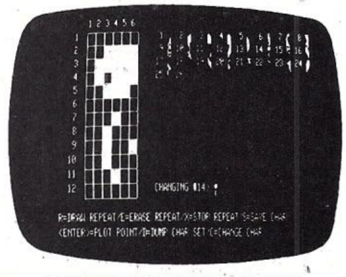
**DEMONSTRATION PROGRAMS**
The 80-GRAFIX board is supplied with a Character Generator software and several demonstration programs.

### FINALLY, AT LAST...

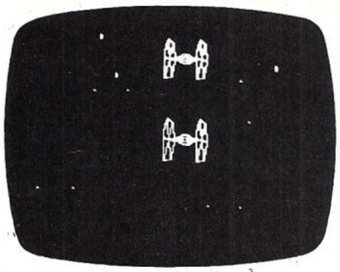
HI- RESOLUTION GRAPHICS is available for your TRS-80 computer system. The 80-GRAFIX board from PROGRAMMA International, Inc. gives your TRS-80 high resolution capability that is greater than the Commodore CBM/PET or even the revered APPLE II.

80-GRAFIX gives the TRS-80 an effective screen of 384X192 pixels, versus the normal 127X192 for the TRS-80, 80X50 for the CBM/PET, or the 280X192 of an APPLE II. As an added feature, 80-GRAFIX offers you lower case characters at no additional cost. Of course, you can also create your own set of up to 64 original characters using the supplied Character Generator software.

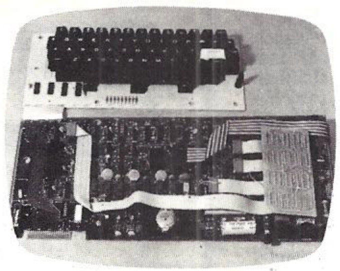
The 80-GRAFIX board is simple to install (note that this voids your Radio Shack warranty), and programming is done through BASIC. 80-GRAFIX opens up a whole new realm of software development and excitement never dreamed of for the TRS-80!



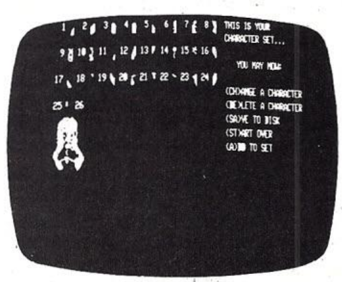
**CHARACTER GENERATOR**
The supplied character generator software allows you to create your own character set of up to 64 original characters.



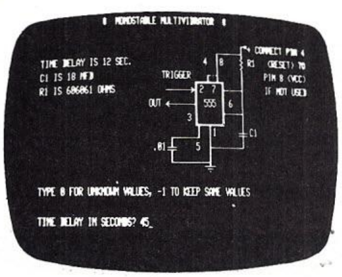
**REAL-TIME GRAPHIC GAMES**
With the 80-GRAFIX board you can write exciting real-time games using BASIC.



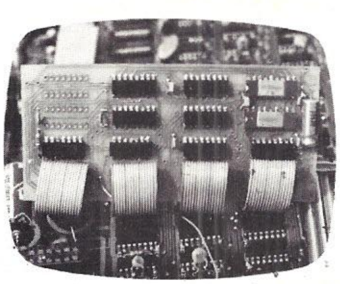
**EASY INSTALLATION**
The 80-GRAFIX board is simple to install and fits inside the TRS-80 case.



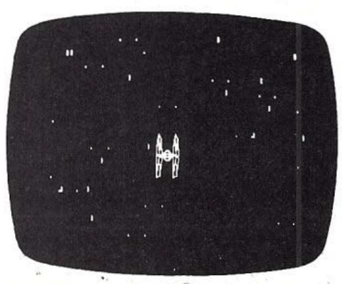

**GRAPHICS GALORE**
The 80-GRAFIX board and the supplied Character Generator allow you to become an artist.




**ELECTRONIC DESIGN**
The 80-GRAFIX board has unlimited application in Electronic design and Education.



**80-GRAFIX HI-RESOLUTION**
Finally, the only means to protect your computer investment is to order an 80-GRAFIX board TODAY!



**EXCITEMENT & FUN**
Open up a new realm of software development with the 80-GRAFIX board.

**Available exclusively through PROGRAMMA at the cost of $149.95**
Please check with us for availability prior to ordering
VISA and MASTERCHARGE accepted
TRS-80 is a registered trademark of the Tandy Corp.

**PROGRAMMA INTERNATIONAL, INC.**
3400 Wilshire Blvd.
Los Angeles, CA 90010
(213) 384-0579 • 384-1116 • 384-1117
✓11

## SPECIAL PRICES

We offer a complete selection of hardware, software, peripherals.

apple
OHIO SCIENTIFIC
VECTOR GRAPHICS
HEWLETT-PACKARD
COMMODORE PET

Immediate shipping.
VISA/MC O.K.
Contact us for a catalog, or our quote.

**Computer Distributors**
**12704 North Freeway**
**Houston, TX 77060**
**(713) 821-2702** ✓152

---

# TRS-80*

## SAVE A BUNDLE

**When you buy your TRS-80™ equipment!**

Use our toll free number to check our price before you buy a TRS-80™ . . . anywhere!

TRS-80 is a trademark of the Radio Shack Division of Tandy Corporation

**full Radio Shack warranty**

**TAB** ✓189
**SALES COMPANY**
1412 WEST FAIRFIELD DR.
P.O. BOX 8098 PENSACOLA FL 32505
904/438-6507
nationwide 1-800-874-1551

---

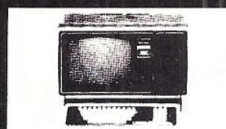
ΩMEGA SALES CO. ✓89

# "WHOLESALE COMPUTER PRICES" DIRECT TO THE PUBLIC

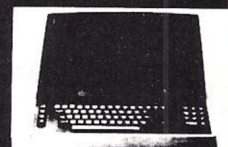
12 Meeting St., Cumberland, R.I. 02864

## PRODUCT SPECIAL OF THE MONTH!!

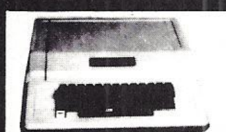
NEC 5510 Printer
$2395
Tractor Feed Option
$180



TRS•80
Model II - $3,500

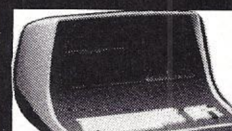


Soroc 120 - $699



Apple II
16K - $1049

Products are NOW IN STOCK AT ΩMEGA Sales Co.



INTERTEC SUPERBRAIN
32K RAM - $2449.00

Atari 800 - $749

For: TRS•80, Apple, CBM, (Interface Included)

Epson TX-80 - $745

## CALL TOLL FREE FOR ΩMEGA'S PRICE!

ΩMEGA OFFERS THE BEST DELIVERY AND PRICE ON:
APPLE • ATARI • TRS-80 MODEL II • INTERTEC • T.I. 810 • HEWLETT-PACKARD-85 • SOROC • COMMODORE • NEC • QUME • CENTRONICS

ΩMEGA sells only factory fresh, top quality merchandise to our customers.
ΩMEGA will try to match any current advertised price with similar purchase conditions.
Before you buy anywhere else - be sure to call ΩMEGA Sales Co.
1-401-722-1027 or

# ΩMEGA TOLL FREE 1-800-556-7586

ΩMEGA ships via UPS, truck, or air. COD's, VISA, Mastercharge accepted.

VISA

# Exploring CT-82 Graphics

*The author leads you on an expedition to uncover the capabilities of this SWTP video terminal.*

*Phil Hughes*
*PO Box 2847*
*Olympia, WA 98507*

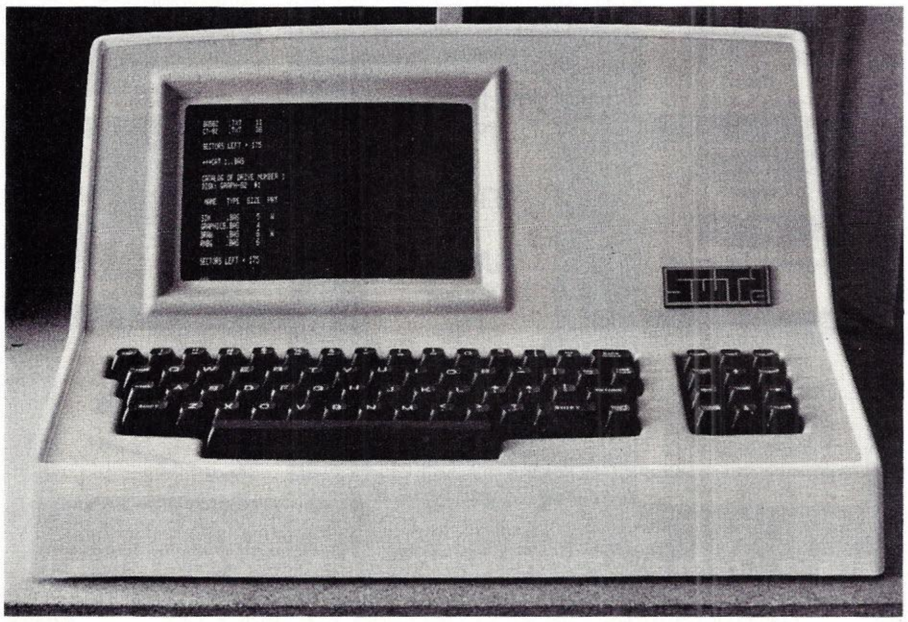


*Photo 1. CT-82 from the outside.*

**CLEAR GRAPHICS DOT**—Turn off a pixel at the specified position.
**INVERT GRAPHICS DOT**—Invert a pixel at the specified graphics position.
**SET GRAPHICS DOT**—Turn on a pixel at the specified graphics position.
**CLEAR GRAPHICS LINE**—Clear the straight line between two specified points.
**INVERT GRAPHICS LINE**—Invert a straight line between two points, i.e., turn off each point that is on and turn on each point that is off.
**SET GRAPHICS LINE**—Draw a straight line between two points.
**PITCH DOWN GRAPHICS SCREEN**—Roll the pixels on the screen downward one pixel position.
**PITCH UP GRAPHICS SCREEN**—Roll the pixels on the screen upward by one pixel position.
**YAW LEFT GRAPHICS SCREEN**—Shift the pixels on the screen left by one pixel position.
**YAW RIGHT GRAPHICS SCREEN**—Shift the pixels on the screen right by one pixel position.

*Fig. 1. CT-82 graphics commands. (Note: A pixel is a picture element, the smallest addressable piece of data on the graphics screen.)*

The CT-82 is a video terminal by Southwest Technical Products Corporation (see Photo 1). Internally, it is based on a 6802 microprocessor and a Motorola 6845 CRT controller IC (see Photo 2). In its normal mode, it can display either 16 or 20 lines of 82 alphanumeric characters each and support scrolling (when the last line on the screen is filled, all lines are rolled up to allow space for a new line). It features other capabilities not found in "dumb" terminals.

Additionally, the CT-82 supports graphics commands that allow you to draw and manipulate dots and lines on the screen. Each of these capabilities is exercised by sending a specific control sequence from the computer to the CT-82. Figure 1 lists the graphics commands supported by the CT-82.

### TEST82

To get the feel of the graphics capabilities, I wrote an assembly-language program, TEST82 (see Listing 1), which allows me to use the set and clear commands for both dots and lines. This program sends the required sequence to put the CT-82 into its graphics mode and then prompts for a command. Figure 2 lists the commands recognized by TEST82. The display illustrated in Photo 3 was created by executing TEST82 and entering the following commands:

| | |
|---|---|
| DL 0 0 183 0 | Top line |
| DL 0 62 183 62 | Bottom line |
| DL 0 0 0 62 | Left side |
| DL 183 0 183 62 | Right side |
| DL 20 0 60 60 | Create |
| DL 60 0 20 60 | X |

Note that all of the assembly-language routines in this article were written using RRMAC from Ed Smith's Software Works. Also, they all reference an "include" file for Flex Interface addresses. (Flex is a disk operating system for the 6800 by Technical Systems Consultants.) Listing 2 is a dummy program that references the include file. This shows all the Flex entry equates and their values.

Internally, TEST82 is easy to follow. When initially entered (at label START), the CT-82 is put in the graphics mode by sending it the string SMSG. Scrolling is also disabled so that entering a carriage return on the command line (bottom line of the screen) will not cause the display to scroll up one line. The code starting at the label NEXTCMD gets a line of user input using the Flex subroutine INBUFF, saves the first input character in accumulator B, saves the second input character in accumulator A and then skips the spaces that should follow the commands.

The command is then analyzed, and control is passed to the appropriate routine—EXIT for the X command, CLEAR for the CS command, DRAW for the D command and ERASE for the E command. If the command is not valid, routine INPERR issues an error message, waits for a carriage return and then returns to the NEXTCMD

| | |
|---|---|
| CS | Clear screen |
| X | Exit |
| DP X1 Y1 | Draw point |
| EP X1 Y1 | Erase point |
| DL X1 Y1 X2 Y2 | Draw line |
| EL X1 Y1 X2 Y2 | Erase line |

Note: the screen is composed of 184 vertical columns of 66 pixels each. These pixels are numbered 0–183 from left to right and 0–65 from top to bottom. The X value in each command is the column number and the Y value is the row number. Rows 63 through 65 are not available as they are used as the command entry line.

*Fig. 2. TEST82 commands.*

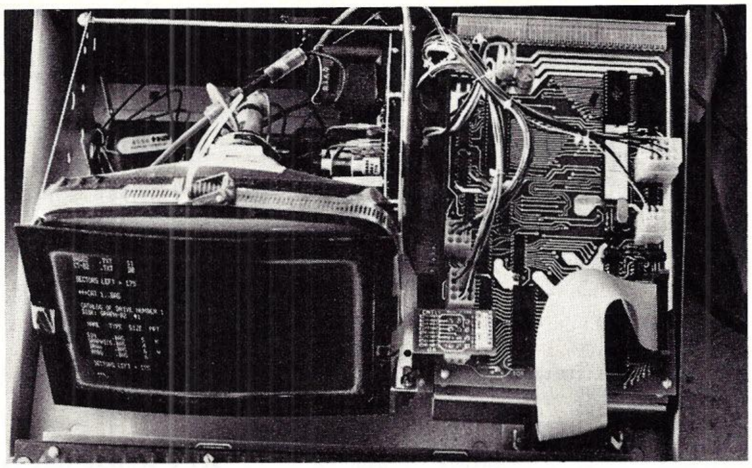


*Photo 2. Inside view of the CT-82.*

*Listing 1. TEST82.*

```
          0000                          NAM    TEST82
          0000                          OPT    XRF,PNT
                             * CT82 GRAPHICS TEST PROGRAM
                             * SSC 8-2-79 V1.14
          0000                          OPT    PNT
                             *
          E1D1               OUTEEE     EQU    X'E1D1'    USED FOR SPECIAL CHARAC
                                                          TERS
          0000               START      EQU    *
   3R     0000 CE 00EB                  LDX    #SMSG
   12 *   0003 BD AD1E                  JSR    PSTRNG
   3R     0006 CE 00F2       NEXTCMD    LDX    #PROMPT
   12 *   0009 BD AD1E                  JSR    PSTRNG
   21 *   000C BD AD1B                  JSR    INBUFF     GET INPUT INTO LINE BUF
                                                          FER
   30 *   000F BD AD27                  JSR    NXTCH      GET FIRST CHAR
   34     0012 24 03 (0017)             BCC    $5
   3R     0014 7E 0040                  JMP    INPERR     BAD INPUT
   2R     0017 16            $5         TAB
   11 *   0018 BD AD27                  JSR    NXTCH      GET POSSIBLE L OR P
   15     001B 36                       PSH A             SAVE
   24 *   001C BD AD27                  JSR    NXTCH      SKIP BLANK
   28     001F 32                       PUL A
   30     0020 C1 43                    CMP B #C'C'       CLEAR?
   34     0022 26 07 (002B)             BNE    $8
   2R     0024 81 53                    CMP A #C'S'       CS - CLEAR SCREEN ?
   6      0026 26 03 (002B)             BNE    $8
   3R     0028 7E 00B3                  JMP    CLEAR
   2R     002B C1 44         $8         CMP B #C'D'       DRAW?
   6      002D 26 03 (0032)             BNE    $10        NO
   3R     002F 7E 0055                  JMP    DRAW
   2R     0032 C1 45         $10        CMP B #C'E'       ERASE?
   6      0034 26 03 (0039)             BNE    $12        NO
   3R     0036 7E 0084                  JMP    ERASE
   2R     0039 C1 58         $12        CMP B #C'X'       EXIT?
   6      003B 26 03 (0040)             BNE    INPERR
   3R     003D 7E 004C                  JMP    EXIT
   3R     0040 CE 00FF       INPERR     LDX    #IEMSG
   12 *   0043 BD AD1E                  JSR    PSTRNG     ERROR MESSAGE
   21 *   0046 BD AD1B                  JSR    INBUFF     WAIT FOR CR
   24     0049 7E 0006                  JMP    NEXTCMD
   3R     004C CE 0125       EXIT       LDX    #RESMSG    RESET CT-82
   12 *   004F BD AD1E                  JSR    PSTRNG
   15     0052 7E AD03                  JMP    WARMS      RETURN TO FLEX
                             *
          0055               DRAW       EQU    *          DRAW POINT OR LINE
   2R     0055 81 50                    CMP A #C'P'
   6      0057 27 07 (0060)             BEQ    DRAW.PT
   2R     0059 81 4C                    CMP A #C'L'
   6      005B 27 12 (006F)             BEQ    DRAW.LN
   3R     005D 7E 0040                  JMP    INPERR     INPUT ERROR
   9R*    0060 BD 00D4       DRAW.PT    JSR    INTWO      GET X AND Y
   14     0063 FE 0149                  LDX    X.AND.Y    GET X AND Y
   20     0066 FF 012F                  STX    DPX        SAVE IN STRING
   23     0069 CE 012D                  LDX    #DP
   26     006C 7E 00B9                  JMP    SEND       SEND IT
                             *
          006F               DRAW.LN    EQU    *
   9R*    006F BD 00C8                  JSR    INFOUR     GET U, V, X, AND Y
   14     0072 FE 0147                  LDX    U.AND.V
   20     0075 FF 0134                  STX    DLU
   25     0078 FE 0149                  LDX    X.AND.Y
   31     007B FF 0136                  STX    DLX
   34     007E CE 0132                  LDX    #DL
   37     0081 7E 00B9                  JMP    SEND
                             *
          0084               ERASE      EQU    *          ERASE LINE OR POINT
   2R     0084 81 50                    CMP A #C'P'       POINT?
   6      0086 27 07 (008F)             BEQ    ERS.PT     YES
   2R     0088 81 4C                    CMP A #C'L'       LINE?
   6      008A 27 12 (009E)             BEQ    ERS.LN     YES
   3R     008C 7E 0040                  JMP    INPERR     ERROR
   9R*    008F BD 00D4       ERS.PT     JSR    INTWO      GET X AND Y
   14     0092 FE 0149                  LDX    X.AND.Y
   20     0095 FF 013B                  STX    EPX
   23     0098 CE 0139                  LDX    #EP
```

```
26    009B 7E 00B9                JMP   SEND
 9R*  009E BD 00C8      ERS.LN    JSR   INFOUR    GET U, V, X, AND Y
14    00A1 FE 0147                LDX   U.AND.V
20    00A4 FF 0140                STX   ELU
25    00A7 FE 0149                LDX   X.AND.Y
31    00AA FF 0142                STX   ELX
34    00AD CE 013E                LDX   #EL
37    00B0 7E 00B9                JMP   SEND
                        *
      00B3              CLEAR     EQU   *         ERASE SCREEN
 3R   00B3 CE 012A                LDX   #CLRMSG
 6    00B6 7E 00B9                JMP   SEND
                        *
      00B9              SEND      EQU   *         SEND STRING AND LOOP TO
                                                   NEXT COMMAND
 5R   00B9 A6 00                  LDA A 0,X       GET CHARACTER
 9    00BB 08                     INX
11    00BC 81 FF                  CMP A #X'FF'    SPECIAL TERMINATOR?
15    00BE 26 03 (00C3)           BNE   $10       NO
 3R   00C0 7E 0006                JMP   NEXTCMD
 9R*  00C3 BD E1D1      $10       JSR   OUTEEE    SEND IT
13    00C6 20 F1 (00B9)           BRA   SEND
                        *
      00C8              INFOUR    EQU   *         GET 4 DECIMAL NUMBERS U
                                                  ,V,X,Y
 9R*  00C8 BD 00E1                JSR   GETDEC    U
14    00CB B7 0147                STA A U
23 *  00CE BD 00E1                JSR   GETDEC    V
28    00D1 B7 0148                STA A V
      00D4              INTWO     EQU   *         GET 2 DECIMAL NUMBERS X
                                                  ,Y
 9R*  00D4 BD 00E1                JSR   GETDEC    X
14    00D7 B7 0149                STA A EX
23 *  00DA BD 00E1                JSR   GETDEC    Y
28    00DD B7 014A                STA A Y
33    00E0 39                     RTS
                        *
      00E1              GETDEC    EQU   *         RETURN VALUE IN A
 9R*  00E1 BD AD48                JSR   INDEC     GET VALUE FROM LINE BUF
                                                  FER
15    00E4 FF 0145                STX   TEMP      SAVE
19    00E7 B6 0146                LDA A TEMP+1    GET L.O. BYTE
24    00EA 39                     RTS
                        *
      00EB 1D16         SMSG      CON   DX'1D16'  PUT IN GRAPHICS MODE
      00ED 0000                   CON   0,0       KILL SOME TIME
      00EF 1E18                   CON   DX'1E18'  DISABLE SCROLLING
      00F1 04                     CON   X'04'
      00F2 0B0015       PROMPT    CON   X'0B',DX'0015' CURSOR POS
      00F5 16                     CON   X'16'     CLEAR TO END OF FRAME
      00F6 434F4D4D414E           CON   C'COMMAND:',X'04'
      00FF 0B0015       IEMSG     CON   X'0B',DX'0015' CURSOR POS
      0102 494E56414C49           CON   C'INVALID INPUT - '
      0112 52455455524E           CON   C'RETURN TO CONTINUE',X'04'
      0125 1C11         RESMSG    CON   DX'1C11'  RESTORE FORMAT 1
      0127 1E08                   CON   DX'1E08'  ENABLE SCROLL
      0129 04                     CON   X'04'
      012A 1C16         CLRMSG    CON   DX'1C16'  CLEAR TO BEGINNING OF F
                                                  RAME
      012C FF                     CON   X'FF'
                        *
      012D 1D13         DP        CON   DX'1D13'  DRAW PT. COMMAND
      012F 0000         DPX       CON   0,0       X,Y
      0131 FF                     CON   X'FF'
                        *
      0132 1D03         DL        CON   DX'1D03'  DRAW LINE COMMAND
      0134 0000         DLU       CON   0,0       U,V
      0136 0000         DLX       CON   0,0       X,Y
      0138 FF                     CON   X'FF'
                        *
      0139 1D14         EP        CON   DX'1D14'  ERASE PT. COMMAND
      013B 0000         EPX       CON   0,0       X,Y
      013D FF                     CON   X'FF'
                        *
      013E 1D04         EL        CON   DX'1D04'  ERASE LINE COMMAND
      0140 0000         ELU       CON   0,0       U,V
      0142 0000         ELX       CON   0,0       X,Y
      0144 FF                     CON   X'FF'
                        *
      0145              TEMP      RMB   2
      0147              U.AND.V   EQU   *
      0147              U         RMB   1
      0148              V         RMB   1
      0149              X.AND.Y   EQU   *
      0149              EX        RMB   1
      014A              Y         RMB   1
                        *
      014B              LAST      ENT
      014B                        END   START
  AD36 ADDBX       66
  AD21 CLASS       59
  00B3 CLEAR      217    159
  012A CLRMSG     261    218
  AD00 COLDS       48
  0132 DL         268    195
  0134 DLU        269    192
  0136 DLX        270    194
  AD4B DOCMND      73
  012D DP         264    186
  012F DPX        265    185
  0055 DRAW       177    162
  006F DRAW.LN    189    181
  0060 DRAW.PT    183    179
  013E EL         277    214
  0140 ELU        278    211
  0142 ELX        279    213
  0139 EP         273    207
  013B EPX        274    206
  0084 ERASE      198    165
  009E ERS.LN     209    202
  008F ERS.PT     204    200
```

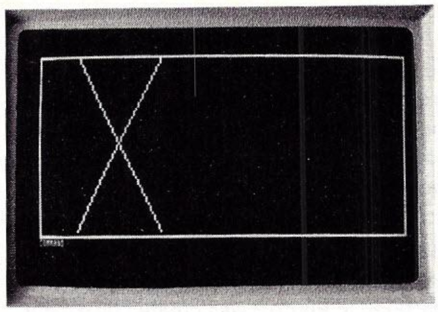


*Photo 3. Sample run of TEST82.*

loop.

Subroutine DRAW determines if the next character is an L or P. An L causes control to be transferred to DRAW.LN, which, in turn, calls INFOUR to get the coordinates and then draws a line. P causes control to be transferred to DRAW.PT, which calls INTWO to get the coordinates and then displays the specified point.

Note that the actual transmission of the control strings is performed by routine SEND. The Flex routine PSTRNG cannot be used because a control string could contain a hexadecimal 04, which acts as a string terminator. SEND uses a hexadecimal FF as the string terminator.

Subroutine ERASE works just like DRAW except the clear control string is sent instead of the set control string. ERS.PT erases a point, and ERS.LN erases a line. To output the control string, all of these routines use SEND, which transfers control back to NEXTCMD.

Subroutine CLEAR sends an erase-to-the-beginning-of-frame command to the CT-82. This is sufficient, since the cursor is on the last line of the screen, which will be changed by NEXTCMD anyway. EXIT restores the CT-82 to screen format 1 (82 × 16 with standard ROM), enables scrolling and returns control to Flex.

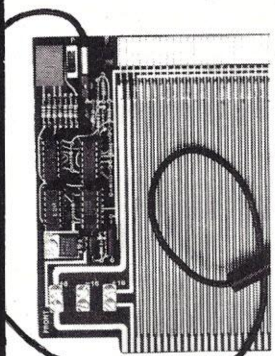
Once I had a chance to play with TEST82 and get a feel for the graphics potential of the CT-82, I decided it was time to interface these capabilities to BASIC. I could either write obscure subroutines in BASIC to send out the control strings or modify TEST82 to be callable from BASIC. I chose the latter approach because it would make the BASIC programs look more readable and be more efficient. Besides, some assembly language would be required anyway to handle the hexadecimal 04 control characters. Listing 3, program BAS82, was the result.

**BAS82**

BAS82 differs from TEST82 in that it picks up its commands from the buffer pointed to by the contents of address 26 (hex) and has an added command, IN, which initializes the CT-82 to graphics mode. Otherwise, the two routines are essentially the

# MULLEN Computer Products

## S-100 EXTENDER/LOGIC PROBE

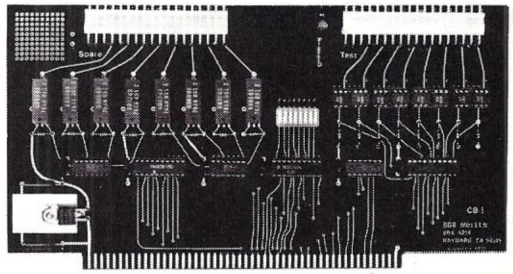


- **New** interlaced ground and signal traces, improves performance, reduces noise, with the new high clock frequency boards
- **New** brighter display, makes this very handy logic probe easier to use
- **New** proposed IEEE buss edge connector label, with all the fine quality documentation you expect with Mullen kits.
- High quality FR-4 board is double sided with plated thru holes and solder-masked for easy kit assembly
- Gold on all mating connector surfaces for better electrical contact
- Formed connector leads for easy scope probe attachment
- Jumper links in power lines makes current measurement and fusing easy
- Large "kluge" area lets you build and test your own circuits

**TB-4 EXTENDER/LOGIC PROBE**
**$59. Kit $79. Assm/tested**

## S-100 CONTROLLER BOARD

- **8 reed relay - OUTPUTS**
- **8 opto-isolated - INPUTS**
- **256 selectable port addresses**



Our S-100 CONTROLLER is used in laboratories, at universities, and in industry, in hundreds of applications, and may be the answer to your control problem. Complete programming and operating instructions included.

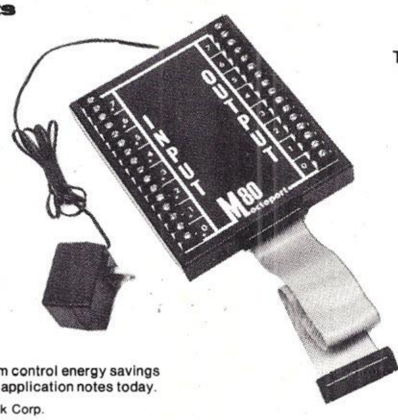
For higher power applications a 500W AC POWER MODULE is available for $15.

**CB-1 CONTROLLER Kit $129.**
**Assm/tested $179.**

**MULLEN Computer Products**

## M-80 CONTROL BOX

**TRS-80* accessory**



The **M-80 OCTOPORT** is a simple to use interface for the **TRS-80 COMPUTER.** You can **control 8 external devices and sense 8 external conditions.** Each output uses a reed relay and each input an opto-isolator to electrically isolate your TRS-80.

One or more controllers can be connected to either the interface connector or the screen printer connector.

**Each OCTOPORT is shipped completely assembled, tested and INCLUDES the interconnector cable, a UL approved power pack, and a 1 year warranty.**

**M-80 OCTOPORT CONTROLLER**
**$159. Assm/tested**

Use your TRS-80, and our M-80 control box to program control energy savings devices at home or in your business. Send for our free application notes today.

*TRS-80 is a trademark of Tandy Radio Shack Corp.

## H8 EXTENDER BOARD

Our HTB-0 lets H8 owners troubleshoot their boards faster and easier. Each board can be extended above the computer for complete access to all circuits and components.

**FEATURES**

- Sturdy 3/32" board
- Molex 25-pin edge connectors with formed leads for easy scope probe attachment
- Jumper links in power lines makes current measurement and fusing easy

**HTB-0 H8 EXTENDER**
**$39. Kit**

*H8 is a trademark of Heath Company.

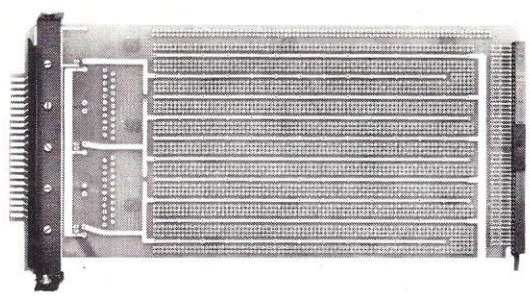
**KLUGE BOARD DESIGNED WITH YOUR PROTOTYPING PROBLEMS IN MIND.**

## H8 PROTOTYPE BOARD

Now available for the Heathkit H8

- Full-sized FR-4 board with heat sink/mounting brackets, buss connectors and polarizing key
- Designed for ease of external cable connection
- All plated thru holes .042" on .1" centers, power and ground traces

**HKB-1 H8 PROTOTYPE BOARD $30. Kit**



PROTOTYPE KIT ACCESSORIES: if you wish to buy any of these accessories for your kit, please list parts and add price to total order. These parts may be ordered at any time, but an additional $1.50 shipping and handling will be charged. . . if ordered separately.

| | | |
|---|---|---|
| PRICES: | $ 5.00 | (1) 44-pin edge/cable connector |
| | $ 2.00 | (1) 5 Volt regulator with (2) 39uF capacitors |
| | $ 1.00 | (1) 25-pin Molex connector (90° male) |
| | $ 1.00 | (1) 25-pin Molex connector (female) |

✓37 **MULLEN COMPUTER PRODUCTS**
**BOX 6214, HAYWARD, CA 94544**
OR PHONE (415) 783-2866 · VISA/MASTERCHARGE ACCEPTED.
INCLUDE $1.50 FOR SHIPPING & HANDLING.
CALIFORNIA RESIDENTS ADD TAX.

**Order Direct or Contact your Local Computer Store.**

```
0149 EX          287   237
004C EXIT        173   168
AC00 F2BACK        8
AC29 F2CCOL       39
AC28 F2COMF       38
AC18 F2CURC       26
AC01 F2DEL         9
AC03 F2DEPC       11
AC07 F2DX         15
AC08 F2EJC        16
AC2D F2ENV        41
AC02 F2EOL        10
AC16 F2ERR        25
AC20 F2ERRT       32
AC0A F2ESC        18
A840 F2FCB        43
AC26 F2FIA        37
AC2F F2FIEF       42
AC24 F2FOA        36
AC23 F2INSW       35
AC11 F2LAST       22
AC14 F2LBP        24
A080 F2LBUF        7
AC1B F2LDAO       29
AC1A F2LINE       28
AC2B F2MEND       40
AC05 F2NULL       13
AC22 F2OTSW       34
AC09 F2PAUS       17
AC19 F2PREC       27
AC0E F2SDR        21
AC0B F2SDRV       19
AC21 F2SIOF       33
AC06 F2TABC       14
AC1D F2TRAN       30
AC1E F2TRNA       31
A100 F2UCA        44
AC12 F2UCTA       23
AC0C F2WDRV       20
AC04 F2WIDC       12
0002 FB2ACT      105
0040 FB2BUF      122
002F FB2CDA      119
001E FB2CUR      115
0003 FB2DRV      106
0013 FB2EDA      111
0001 FB2ESB      104
000C FB2EXT      108
000F FB2FAT      109
0032 FB2FDP      120
001C FB2FLP      114
0000 FB2FNC      103
0022 FB2INX      116
0004 FB2NAM      107
0024 FB2NWB      118
0023 FB2RDX      117
003B FB2SCF      121
0011 FB2SDA      110
0015 FB2SIZ      112
0017 FB2SMI      113
0008 FEXT.BAC    133
0005 FEXT.BAK    130
0003 FEXT.BAS    128
0000 FEXT.BIN    125
0002 FEXT.CMD    127
0007 FEXT.DAT    132
0009 FEXT.DIR    134
000B FEXT.OUT    136
000A FEXT.PRT    135
0006 FEXT.SCR    131
0004 FEXT.SYS    129
0001 FEXT.TXT    126
0000 FL2IO        80
0016 FM2BOR       99
0004 FM2CLF       84
000C FM2DLF       91
0014 FM2FND       97
0007 FM2GIR       87
0011 FM2GRB       95
000F FM2NSS       93
0006 FM2OPD       86
0001 FM2OPR       81
0003 FM2OPU       83
0002 FM2OPW       82
0010 FM2OSI       94
0008 FM2PIR       88
0015 FM2POS       98
0012 FM2PRB       96
000D FM2RNF       92
0009 FM2RSS       89
0005 FM2RWF       85
000A FM2WSS       90
B400 FMINIT       74
B406 FMS          76
B403 FMSCLS       75
AD15 GETCHR       55
00E1 GETDEC      242   231   233   236   238
AD2D GETFIL       63
AD42 GETHEX       70
00FF IEMSG       255   169
AD1B INBUFF       57   146   171
AD09 INCH         51
AD0C INCH2        52
AD48 INDEC        72   243
00C8 INFOUR      230   190   209
0040 INPERR      169   149   167   182   203
00D4 INTWO       235   183   204
014B LAST        290
AD30 LOAD         64
0006 NEXTCMD     144   172   226
AD27 NXTCH        61   147   151   153
AD45 OUTADR       71
AD0F OUTCH        53
AD12 OUTCH2       54
AD39 OUTDEC       67
E1D1 OUTEEE      140   227
AD3C OUTHEX       68
AD24 PCRLF        60
00F2 PROMPT      252   144
AD1E PSTRNG       58   143   145   170   174
AD18 PUTCHR       56
AD06 RENTER       50
0125 RESMSG      258   173
AD3F RPTERR       69
AD2A RSTRIO       62
00B9 SEND        221   187   196   208   215   219   228
AD33 SETEXT       65
00EB SMSG        248   142
0000 START       141   291
0145 TEMP        282   244   245
0147 U           284   232
0147 U.AND.V     283   191   210
0148 V           285   234
AD03 WARMS        49   175
0149 X.AND.Y     286   184   193   205   212
014A Y           288   239
```

*Listing 2. Dummy program that shows all the Flex entry equates and their values.*

```
0000           NAM    DUMMY      - FOR LIST OF FLEXEQU.I
                                 NC
               *:0.FLEXEQU.INC
A080    F2LBUF   EQU    X'A080'    LINE BUFFER
AC00    F2BACK   EQU    X'AC00'    BACKSPACE CHARACTER
AC01    F2DEL    EQU    X'AC01'    DELETE CHARACTER
AC02    F2EOL    EQU    X'AC02'    END OF LINE CHAR.
AC03    F2DEPC   EQU    X'AC03'    PAGE DEPTH COUNT
AC04    F2WIDC   EQU    X'AC04'    WIDTH COUNT
AC05    F2NULL   EQU    X'AC05'    NULL COUNT
AC06    F2TABC   EQU    X'AC06'    TAB CHARACTER
AC07    F2DX     EQU    X'AC07'    DUPLEX MODE
AC08    F2EJC    EQU    X'AC08'    EJECT COUNT
AC09    F2PAUS   EQU    X'AC09'    PAUSE CONTROL
AC0A    F2ESC    EQU    X'AC0A'    ESCAPE CHARACTER
AC0B    F2SDRV   EQU    X'AC0B'    SYSTEM DRIVE NUMBER
AC0C    F2WDRV   EQU    X'AC0C'    WORKING DRIVE
AC0E    F2SDR    EQU    X'AC0E'    3 BYTE DATE REGISTER
AC11    F2LAST   EQU    X'AC11'    LAST TERMINATOR
AC12    F2UCTA   EQU    X'AC12'    USER COMMAND TABLE ADDR
                                   ESS
AC14    F2LBP    EQU    X'AC14'    LINE BUFFER POINTER
AC16    F2ERR    EQU    X'AC16'    ESCAPE RETURN REGISTER
AC18    F2CURC   EQU    X'AC18'    CURRENT CHARACTER
AC19    F2PREC   EQU    X'AC19'    PREVIOUS CHARACTER
AC1A    F2LINE   EQU    X'AC1A'    CURRENT LINE NUMBER
AC1B    F2LDAO   EQU    X'AC1B'    LOADER ADDRESS OFFSET
AC1D    F2TRAN   EQU    X'AC1D'    TRANSFER FLAG
AC1E    F2TRNA   EQU    X'AC1E'    TRANSFER ADDRESS
AC20    F2ERRT   EQU    X'AC20'    ERROR TYPE
AC21    F2SIOF   EQU    X'AC21'    SPCIAL I/O FLAG
AC22    F2OTSW   EQU    X'AC22'    OUTPUT SWITCH
AC23    F2INSW   EQU    X'AC23'    INPUT SWITCH
```

```
AC24        F2FOA     EQU   X'AC24'   FILE OUTPUT ADDRESS
AC26        F2FIA     EQU   X'AC26'   FILE INPUT ADDRESS
AC28        F2COMF    EQU   X'AC28'   COMMAND FLAG
AC29        F2CCOL    EQU   X'AC29'   CURRENT OUTPUT COLUMN
AC2B        F2MEND    EQU   X'AC2B'   MEMORY END
AC2D        F2ENV     EQU   X'AC2D'   ERROR NAME VECTOR
AC2F        F2FIEF    EQU   X'AC2F'   FILE INPUT ECHO FLAG
A840        F2FCB     EQU   X'A840'   SYSTEM FCB
A100        F2UCA     EQU   X'A100'   UTILITY COMMAND AREA
            *
            * FLEX SUBROUTINES
            *
AD00        COLDS     EQU   X'AD00'   COLDSTART ENTRY
AD03        WARMS     EQU   X'AD03'   WARM START ENTRY
AD06        RENTER    EQU   X'AD06'   MAIN LOOP RE-ENTRY
AD09        INCH      EQU   X'AD09'   INPUT CHAR
AD0C        INCH2     EQU   X'AD0C'   INPUT CHAR
AD0F        OUTCH     EQU   X'AD0F'   OUTPUT CHAR
AD12        OUTCH2    EQU   X'AD12'   OUTPUT CHAR
AD15        GETCHR    EQU   X'AD15'   PREFERRED GET CHAR
AD18        PUTCHR    EQU   X'AD18'   PREFERRED PUT CHAR
AD1B        INBUFF    EQU   X'AD1B'   INPUT TO LINE BUFFER
AD1E        PSTRNG    EQU   X'AD1E'   PRINT STRING
AD21        CLASS     EQU   X'AD21'   CLASSIFY CHARACTER
AD24        PCRLF     EQU   X'AD24'   PRINT CR, LF
AD27        NXTCH     EQU   X'AD27'   NEXT CHARACTER
AD2A        RSTRIO    EQU   X'AD2A'   RESTORE I/O VECTORS
AD2D        GETFIL    EQU   X'AD2D'   PARSE FILE SPEC.
AD30        LOAD      EQU   X'AD30'   FILE LOADER
AD33        SETEXT    EQU   X'AD33'   SET EXTENSION
AD36        ADDBX     EQU   X'AD36'   ADD ACC-B TO X
AD39        OUTDEC    EQU   X'AD39'   OUTPUT DECIMAL NUMBER
AD3C        OUTHEX    EQU   X'AD3C'   OUTPUT HEX CHARACTER
AD3F        RPTERR    EQU   X'AD3F'   REPORT ERROR
AD42        GETHEX    EQU   X'AD42'   GET HEX NUMBER
AD45        OUTADR    EQU   X'AD45'   OUTPUT HEX ADDRESS
AD48        INDEC     EQU   X'AD48'   INPUT DECIMAL NUMBER
AD4B        DOCMND    EQU   X'AD4B'   CALL DOS
B400        FMINIT    EQU   X'B400'   FMS INITIALIZATION
B403        FMSCLS    EQU   X'B403'   FMS CLOSE
B406        FMS       EQU   X'B406'   FMS CALL
            *
            * FMS COMMANDS
            *
0000        FL2IO     EQU   X'0'      READ/WRITE NEXT BYTE
0001        FM2OPR    EQU   X'1'      OPEN FOR READ
0002        FM2OPW    EQU   X'2'      OPEN FOR WRITE
0003        FM2OPU    EQU   X'3'      OPEN FOR UPDATE
0004        FM2CLF    EQU   X'4'      CLOSE FILE
0005        FM2RWF    EQU   X'5'      REWIND FILE
0006        FM2OPD    EQU   X'6'      OPEN DIRECTORY
0007        FM2GIR    EQU   X'7'      GET INFORMATION RECORD
0008        FM2PIR    EQU   X'8'      PUT INFORMATION RECORD
0009        FM2RSS    EQU   X'9'      READ SINGLE SECTOR
000A        FM2WSS    EQU   X'A'      WRITE SINGLE SECTOR
000C        FM2DLF    EQU   X'C'      DELETE FILE
000D        FM2RNF    EQU   X'D'      RENAME FILE
000F        FM2NSS    EQU   X'F'      NEXT SEQUENTIAL SECTOR
0010        FM2OSI    EQU   X'10'     OPEN SYSTEM INFORMATION
                                       RECORD
0011        FM2GRB    EQU   X'11'     GET RANDOM BYTE FROM SE
                                      CTOR
0012        FM2PRB    EQU   X'12'     PUT RANDOM BYTE IN SECT
                                      OR
0014        FM2FND    EQU   X'14'     FIND NEXT DRIVE
0015        FM2POS    EQU   X'15'     POSITION BY RECORD
0016        FM2BOR    EQU   X'16'     BACKUP 1 RECORD
            *
            * FILE CONTROL BLOCK SPECIFICATIONS
            *
0000        FB2FNC    EQU   0         FMS COMMAND
0001        FB2ESB    EQU   1         ERROR STATUS
0002        FB2ACT    EQU   2         ACTIVITY STATUS
0003        FB2DRV    EQU   3         DRIVE NUMBER
0004        FB2NAM    EQU   4         FILE NAME (8 BYTES)
000C        FB2EXT    EQU   12        EXTENSION (3 BYTES)
000F        FB2FAT    EQU   15        FILE ATTRIBUTES
0011        FB2SDA    EQU   17        STARTING DISK ADDRESS
0013        FB2EDA    EQU   19        ENDING DISK ADDRESS
0015        FB2SIZ    EQU   21        FILE SIZE
0017        FB2SMI    EQU   23        FILE SECTOR MAP INDICAT
                                      OR
001C        FB2FLP    EQU   28        FCB LIST POINTER
001E        FB2CUR    EQU   30        CURRENT POSITION
0022        FB2INX    EQU   34        DATA INDEX
0023        FB2RDX    EQU   35        RANDOM INDEX
0024        FB2NWB    EQU   36        NAME WORK BUFFER
002F        FB2CDA    EQU   47        CURRENT DIRECTORY ADDRE
                                      SS
0032        FB2FDP    EQU   50        FIRST DELETED DIRECTORY
                                       POINTER
003B        FB2SCF    EQU   59        SPACE COMPRESSION FLAG
0040        FB2BUF    EQU   64        SECTOR BUFFER
            *
            * FILE EXTENSIONS
0000        FEXT.BIN  EQU   0
0001        FEXT.TXT  EQU   1
0002        FEXT.CMD  EQU   2
0003        FEXT.BAS  EQU   3
0004        FEXT.SYS  EQU   4
0005        FEXT.BAK  EQU   5
0006        FEXT.SCR  EQU   6
0007        FEXT.DAT  EQU   7
0008        FEXT.BAC  EQU   8
0009        FEXT.DIR  EQU   9
000A        FEXT.PRT  EQU   10
000B        FEXT.OUT  EQU   11
            *
0000                  END
```



✓ Reader Service index—page 241

same. I load BAS82 at address C800, where I have some free memory. The program must be loaded somewhere that BASIC doesn't know about so it isn't wiped out.

Next, I wrote a set of subroutines in BASIC that called the different functions in BAS82. These subroutines appear in lines 9000 through 9630 in Listings 4, 5 and 6. Lines 9000 through 9070 comprise the initialization subroutine. They call the system loader to load BAS82, set the BAS82 entry point in the location BASIC uses for the address of the USR function, call BAS82 with a parameter value of "IN" and turn off the cursor display.

Lines 9100 through 9140 draw a point. Note that X is the column coordinate with valid values from 0 to 183. Y is the row coordinate with valid values from 0 to 65.

The draw-line subroutine in lines 9200 through 9240 uses variables U and V as the column and row coordinates of one end of the line, and variables X and Y as the column and row coordinates of the other end. Again, BAS82 is called to transmit the appropriate control string to the CT-82.

The clear subroutine in lines 9300 through 9340 sends a clear screen command. The exit subroutine in lines 9400 through 9450 restores the CT-82 to non-graphics mode and turns on the cursor.

The erase line subroutine in lines 9500 through 9540 and the erase point subroutine in lines 9600 through 9630 work the same as their equivalent draw subroutines.

Using these subroutines, I wrote three BASIC programs. The first, Random Draw/Erase Line (Listing 4), generates coordinates randomly, draws a line, generates another set of random coordinates, erases a line and then repeats this loop the specified number of times. Although this is much more interesting to watch when it is running than after it has stopped, Photo 4 shows how it appears.

Listing 5 shows a more practical program that draws a sine curve (see Photo 5). It essentially uses the equation Y = SIN(X). Lines 310 and 320 look strange but they handle the scaling for the screen dimensions; lower values of Y will cause the curve to move up on the screen. Also, line 420 positions the cursor and labels the graph.

The third program (Listing 6) generates a bar graph (see Photo 6). This graph shows the frequency distribution of the function x = RND(0)*20 for 600 trials. In other words, it shows how random the random number generator is by trying it 600 times and graphically displaying the result. The bar-graphing technique is useful in displaying data for easy comparison.

This is currently where I am exploring the capabilities of the CT-82. Using the basic concepts presented in this article, you will be able to discover even more. ■

*Listing 3. BAS82.*

```
       0000                            NAM    BAS82
                             * CT82-BASIC GRAPHICS INTERFACE
                             * SSC 8-16-79 V1.7
       0000                            OPT    PNT,XRF
                             *
       E1D1                  OUTEEE    EQU    X'E1D1'   USED FOR FUNNY CHARACTE
                                                        RS
       0026                  PASS      EQU    X'26'     ADDRESS PASSED BY BASIC

                             *
       0000                  BAS.82    ENT
  4R   0000 DE 26                      LDX    PASS      DESCRIPTOR ADDRESS
 10    0002 EE 00                      LDX    0,X       STRING ADDRESS (PACKET)

 16    0004 FF 0134                    STX    PKTADR    SAVE
 21    0007 E6 00                      LDA B 0,X        GET
 26    0009 A6 01                      LDA A 1,X        FUNCTION
                             *
 28    000B C1 49                      CMP B #C'I'      INITIALIZE?
 32    000D 26 07 (0016)               BNE    $5
  2R   000F 81 4E                      CMP A #C'N'
  6    0011 26 03 (0016)               BNE    $5
  3R   0013 7E 00B7                    JMP    INIT      YES - INITIALIZE CT82
       0016                  $5        NUL
  2R   0016 C1 43                      CMP B #C'C'      CLEAR?
  6    0018 26 07 (0021)               BNE    $8
  2R   001A 81 53                      CMP A #C'S'      CS - CLEAR SCREEN ?
  6    001C 26 03 (0021)               BNE    $8
  3R   001E 7E 00B1                    JMP    CLEAR
  2R   0021 C1 44            $8        CMP B #C'D'      DRAW?
  6    0023 26 03 (0028)               BNE    $10       NO
  3R   0025 7E 0053                    JMP    DRAW
  2R   0028 C1 45            $10       CMP B #C'E'      ERASE?
  6    002A 26 03 (002F)               BNE    $12       NO
  3R   002C 7E 0082                    JMP    ERASE
  2R   002F C1 58            $12       CMP B #C'X'      EXIT?
  6    0031 26 03 (0036)               BNE    INPERR
  3R   0033 7E 0040                    JMP    EXIT
  3R   0036 CE 00ED          INPERR    LDX    #IEMSG
 12 *  0039 BD 00BD                    JSR    SEND      ERROR MESSAGE
 21 *  003C BD AD1B                    JSR    INBUFF    WAIT FOR CR
 26    003F 39                         RTS
  3R   0040 CE 0113          EXIT      LDX    #RESMSG.1 RESET CT-82
 12 *  0043 BD 00BD                    JSR    SEND
 15    0046 CE FFFF                    LDX    #X'FFFF'  WAIT A WHILE
  4R   0049 09               $5        DEX
  8    004A 26 FD (0049)               BNE    $5
  3R   004C CE 0116                    LDX    #RESMSG.2 RESTORE SCROLLING
 12 *  004F BD 00BD                    JSR    SEND
 17    0052 39                         RTS              RETURN TO BASIC
                             *
       0053                  DRAW      EQU    *         DRAW POINT OR LINE
  2R   0053 81 50                      CMP A #C'P'
  6    0055 27 07 (005E)               BEQ    DRAW.PT
  2R   0057 81 4C                      CMP A #C'L'
  6    0059 27 12 (006D)               BEQ    DRAW.LN
  3R   005B 7E 0036                    JMP    INPERR    INPUT ERROR
  9R*  005E BD 00D8          DRAW.PT   JSR    INTWO     GET X AND Y
 14    0061 FE 013A                    LDX    X.AND.Y   GET X AND Y
 20    0064 FF 011E                    STX    DPX       SAVE IN STRING
 23    0067 CE 011C                    LDX    #DP
 26    006A 7E 00BD                    JMP    SEND      SEND IT
                             *
       006D                  DRAW.LN   EQU    *
  9R*  006D BD 00CB                    JSR    INFOUR    GET U, V, X, AND Y
 14    0070 FE 0138                    LDX    U.AND.V
 20    0073 FF 0123                    STX    DLU
 25    0076 FE 013A                    LDX    X.AND.Y
 31    0079 FF 0125                    STX    DLX
 34    007C CE 0121                    LDX    #DL
 37    007F 7E 00BD                    JMP    SEND
                             *
       0082                  ERASE     EQU    *         ERASE LINE OR POINT
  2R   0082 81 50                      CMP A #C'P'      POINT?
  6    0084 27 07 (008D)               BEQ    ERS.PT    YES
  2R   0086 81 4C                      CMP A #C'L'      LINE?
  6    0088 27 12 (009C)               BEQ    ERS.LN    YES
  3R   008A 7E 0036                    JMP    INPERR    ERROR
  9R*  008D BD 00D8          ERS.PT    JSR    INTWO     GET X AND Y
 14    0090 FE 013A                    LDX    X.AND.Y
 20    0093 FF 012A                    STX    EPX
 23    0096 CE 0128                    LDX    #EP
 26    0099 7E 00BD                    JMP    SEND
  9R*  009C BD 00CB          ERS.LN    JSR    INFOUR    GET U, V, X, AND Y
 14    009F FE 0138                    LDX    U.AND.V
 20    00A2 FF 012F                    STX    ELU
 25    00A5 FE 013A                    LDX    X.AND.Y
 31    00A8 FF 0131                    STX    ELX
 34    00AB CE 012D                    LDX    #EL
 37    00AE 7E 00BD                    JMP    SEND
                             *
       00B1                  CLEAR     EQU    *         ERASE SCREEN
  3R   00B1 CE 0119                    LDX    #CLRMSG
  6    00B4 7E 00BD                    JMP    SEND
                             *
       00B7                  INIT      EQU    *         INITIALIZE CT-82
  3R   00B7 CE 00E6                    LDX    #SMSG
  6    00BA 7E 00BD                    JMP    SEND
                             *
       00BD                  SEND      EQU    *         SEND STRING AND LOOP TO
                                                         NEXT COMMAND
  5R   00BD A6 00                      LDA A 0,X        GET CHARACTER
  9    00BF 08                         INX
 11    00C0 81 FF                      CMP A #X'FF'     SPECIAL TERMINATOR?
 15    00C2 26 01 (00C5)               BNE    $10       NO
 20    00C4 39                         RTS
```

```
  9R* 00C5 BD E1D1         $10        JSR    OUTEEE     SEND IT
  13  00C8 20 F3 (00BD)               BRA    SEND
  18  00CA 39                         RTS
                           *
      00CB                 INFOUR     EQU    *          GET 4 DECIMAL NUMBERS U
                                                        ,V,X,Y
   5R 00CB FE 0134                    LDX    PKTADR     GET PACKET ADDRESS
  10  00CE A6 04                      LDA A 4,X         GET U
  15  00D0 B7 0138                    STA A U
  20  00D3 A6 05                      LDA A 5,X         GET V
  25  00D5 B7 0139                    STA A V
      00D8                 INTWO      EQU    *          GET 2 DECIMAL NUMBERS X
                                                        ,Y
   5R 00D8 FE 0134                    LDX    PKTADR     GET PACKET ADDRESS
  10  00DB A6 02                      LDA A 2,X         GET X
  15  00DD B7 013A                    STA A EX
  20  00E0 A6 03                      LDA A 3,X         GET Y
  25  00E2 B7 013B                    STA A Y
  30  00E5 39                         RTS
                           *
                           *
      00E6 1D16            SMSG       CON    DX'1D16'   PUT IN GRAPHICS MODE
      00E8 0000                       CON    0,0        KILL SOME TIME
      00EA 1E18                       CON    DX'1E18'   DISABLE SCROLLING
      00EC FF                         CON    X'FF'
      00ED 0B0015          IEMSG      CON    X'0B',DX'0015' CURSOR POS
      00F0 494E56414C49               CON    C'INVALID INPUT - '
      0100 52455455524E               CON    C'RETURN TO CONTINUE',X'FF'
      0113 1C11            RESMSG.1   CON    DX'1C11'   RESTORE FORMAT 1
      0115 FF                         CON    X'FF'
      0116 1E08            RESMSG.2   CON    DX'1E08'   ENABLE SCROLL
      0118 FF                         CON    X'FF'
      0119 1016            CLRMSG     CON    DX'1016'   CLEAR  SCREEN
      011B FF                         CON    X'FF'
                           *
      011C 1D13            DP         CON    DX'1D13'   DRAW PT. COMMAND
      011E 0000            DPX        CON    0,0        X,Y
      0120 FF                         CON    X'FF'
                           *
      0121 1D03            DL         CON    DX'1D03'   DRAW LINE COMMAND
      0123 0000            DLU        CON    0,0        U,V
      0125 0000            DLX        CON    0,0        X,Y
      0127 FF                         CON    X'FF'
                           *
      0128 1D14            EP         CON    DX'1D14'   ERASE PT. COMMAND
      012A 0000            EPX        CON    0,0        X,Y
      012C FF                         CON    X'FF'
                           *
      012D 1D04            EL         CON    DX'1D04'   ERASE LINE COMMAND
      012F 0000            ELU        CON    0,0        U,V
      0131 0000            ELX        CON    0,0        X,Y
      0133 FF                         CON    X'FF'
                           *
      0134                 PKTADR     RMB    2          PACKET ADDRESS SAVE
      0136                 TEMP       RMB    2
      0138                 U.AND.V    EQU    *
      0138                 U          RMB    1
      0139                 V          RMB    1
      013A                 X.AND.Y    EQU    *
      013A                 EX         RMB    1
      013B                 Y          RMB    1
                           *
      013C                 LAST       ENT
      013C                            END
```

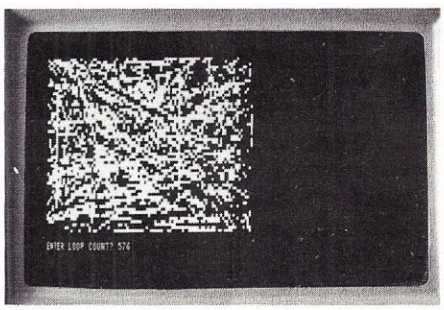



*Photo 4. Sample output of random draw/erase line program.*

*Listing 4. Random draw/erase line program.*

```
10 REM RANDOM DRAW/ERASE LINE
100 GOSUB 9000
200 GOSUB 9300
220 GOTO 1000
250 FOR I=1 TO L
300 U=RND(0)*100
310 V=RND(0)*60
320 X=RND(0)*100
330 Y=RND(0)*60
400 GOSUB 9200 : REM DRAW A LINE
500 U=RND(0)*100
510 V=RND(0)*60
529 X=RND(0)*100
530 Y=RND(0)*60
```

```
550 GOSUB 9500 : REM ERASE A LINE
900 NEXT I
1000 PRINT CHR$(11);CHR$(0);CHR$(21);
1010 PRINT CHR$(6):REM ERASE TO EOL
1020 INPUT "ENTER LOOP COUNT";L
1040 IF L>0 THEN 250
1060 GOSUB 9400
1100 END
9000 REM INITIALIZE CT-82 GOSUB
9010 EXEC,"GET 1.BAS82.BIN"
9020 POKE HEX("24"),HEX("C8"):POKE HEX("25"),HEX("00")
9030 ZZ$="IN"
9040 Z=USR(PTR(ZZ$))
9050 PRINT CHR$(30);CHR$(21):REM TURN OFF CURSOR
9060 RETURN
9070 REM ***
9100 REM DRAW POINT (X,Y) GOSUB
9110 ZZ$="DP"+CHR$(X)+CHR$(Y)
9120 Z=USR(PTR(ZZ$))
9130 RETURN
9140 REM ***
9200 REM DRAW LINE (U,V,X,Y) GOSUB
9210 ZZ$="DL"+CHR$(U)+CHR$(V)+CHR$(X)+CHR$(Y)
9220 Z=USR(PTR(ZZ$))
9230 RETURN
9240 REM **
9300 REM CLEAR GOSUB
9310 ZZ$="CS"
9320 Z=USR(PTR(ZZ$))
9330 RETURN
9340 REM ***
9400 REM EXIT GOSUB
9410 ZZ$="X"
9420 Z=USR(PTR(ZZ$))
9430 PRINT CHR$(30);CHR$(5)
9440 RETURN
9450 REM ***
9500 REM ERASE LINE (U,V,X,Y) GOSUB
9510 ZZ$="EL"+CHR$(U)+CHR$(V)+CHR$(X)+CHR$(Y)
9520 Z=USR(PTR(ZZ$))
9530 RETURN
9540 REM ***
9600 REM ERASE POINT (X,Y) GOSUB
9610 ZZ$="EP"+CHR$(X)+CHR$(Y)
9620 Z=USR(PTR(ZZ$))
9630 RETURN
```

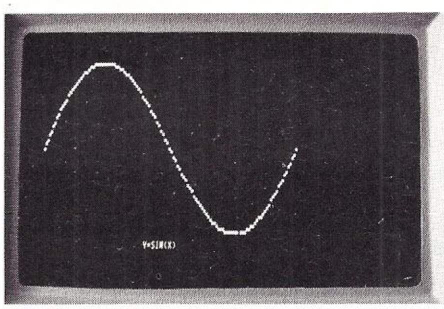



*Photo 5. Sample output from sine curve program.*

*Listing 5. Draw sine curve program.*

```
10 REM DRAW SINE CURVE
100 GOSUB 9000
300 FOR I=0 TO 2*PI STEP .05
310 X=20*I
320 Y=30*(1-SIN(I))+1
350 GOSUB 9100
400 NEXT I
420 PRINT CHR$(11);CHR$(25);CHR$(21);"Y=SIN(X)";
500 INPUT #0,A$
600 GOSUB 9400
700 END
9000 REM INITIALIZE CT-82 GOSUB
9010 EXEC,"GET 1.BAS82.BIN"
9020 POKE HEX("24"),HEX("C8"):POKE HEX("25"),HEX("00")
9030 ZZ$="IN"
9040 Z=USR(PTR(ZZ$))
9050 PRINT CHR$(30);CHR$(21):REM TURN OFF CURSOR
9060 RETURN
9070 REM ***
9100 REM DRAW POINT (X,Y) GOSUB
9110 ZZ$="DP"+CHR$(X)+CHR$(Y)
9120 Z=USR(PTR(ZZ$))
9130 RETURN
9140 REM ***
9200 REM DRAW LINE (U,V,X,Y) GOSUB
9210 ZZ$="DL"+CHR$(U)+CHR$(V)+CHR$(X)+CHR$(Y)
9220 Z=USR(PTR(ZZ$))
9230 RETURN
9240 REM **
9300 REM CLEAR GOSUB
9310 ZZ$="CS"
9320 Z=USR(PTR(ZZ$))
```

```
9330 RETURN
9340 REM ***
9400 REM EXIT GOSUB
9410 ZZ$="X"
9420 Z=USR(PTR(ZZ$))
9430 PRINT CHR$(30);CHR$(5)
9440 RETURN
9450 REM ***
9500 REM ERASE LINE (U,V,X,Y) GOSUB
9510 ZZ$="EL"+CHR$(U)+CHR$(V)+CHR$(X)+CHR$(Y)
9520 Z=USR(PTR(ZZ$))
9530 RETURN
9540 REM ***
9600 REM ERASE POINT (X,Y) GOSUB
9610 ZZ$="EP"+CHR$(X)+CHR$(Y)
9620 Z=USR(PTR(ZZ$))
9630 RETURN
```

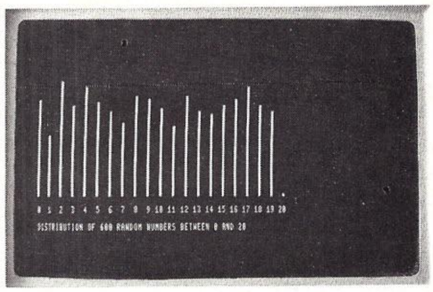



*Photo 6. Output sample from bar graph program.*

```
10 REM BAR GRAPH OF RANDOM NUMBER DISTRIBUTION
90 DIM V(20)
100 GOSUB 9000
200 FOR I=1 TO 600
210 X=RND(0)*20
220 V(X)=V(X)+1
230 NEXT I
300 V=50
310 FOR I=0 TO 20
320 X=6*I
330 Y=50-V(I)
340 U=X
350 GOSUB 9200
360 NEXT I
400 PRINT CHR$(11);CHR$(0);CHR$(18);
410 PRINT "0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20"
420 PRINT
430 PRINT "DISTRIBUTION OF 600 RANDOM NUMBERS BETWEEN 0 AND 20"
500 INPUT #0,A$
600 GOSUB 9400
610 END
9000 REM INITIALIZE CT-82 GOSUB
9010 EXEC,"GET 1.BAS82.BIN"
9020 POKE HEX("24"),HEX("C8"):POKE HEX("25"),HEX("00")
9030 ZZ$="IN"
9040 Z=USR(PTR(ZZ$))
9050 PRINT CHR$(30);CHR$(21):REM TURN OFF CURSOR
9060 RETURN
9070 REM ***
9100 REM DRAW POINT (X,Y) GOSUB
9110 ZZ$="DP"+CHR$(X)+CHR$(Y)
9120 Z=USR(PTR(ZZ$))
9130 RETURN
9140 REM ***
9200 REM DRAW LINE (U,V,X,Y) GOSUB
9210 ZZ$="DL"+CHR$(U)+CHR$(V)+CHR$(X)+CHR$(Y)
9220 Z=USR(PTR(ZZ$))
9230 RETURN
9240 REM **
9300 REM CLEAR GOSUB
9310 ZZ$="CS"
9320 Z=USR(PTR(ZZ$))
9330 RETURN
9340 REM ***
9400 REM EXIT GOSUB
9410 ZZ$="X"
9420 Z=USR(PTR(ZZ$))
9430 PRINT CHR$(30);CHR$(5)
9440 RETURN
9450 REM ***
9500 REM ERASE LINE (U,V,X,Y) GOSUB
9510 ZZ$="EL"+CHR$(U)+CHR$(V)+CHR$(X)+CHR$(Y)
9520 Z=USR(PTR(ZZ$))
9530 RETURN
9540 REM ***
9600 REM ERASE POINT (X,Y) GOSUB
9610 ZZ$="EP"+CHR$(X)+CHR$(Y)
9620 Z=USR(PTR(ZZ$))
9630 RETURN
```

*Listing 6. Bar graph of random number distribution program.*

**NEW! TPM* for TRS-80 Model II**
**NEW! System/6 Package**
Computer Design Labs

# Z80* Disk Software

*We have acquired the rights to all TDL software (& hardware). TDL software has long had the reputation of being the best in the industry. Computer Design Labs will continue to maintain, evolve and add to this superior line of quality software.*

*— Carl Galletti and Roger Amidon, owners.*

Software with Manual/Manual Alone

**All of the software below is available on any of the following media for operation with a Z80 CPU using the CP/M* or similar type disk operating system (such as our own TPM*).**

**for TRS-80* CP/M (Model I or II)**
**for 8" CP/M (soft sectored single density)**
**for 5¼" CP/M (soft sectored single density)**
**for 5¼" North Star CP/M (single density)**
**for 5¼" North Star CP/M (double density)**

### BASIC I

A powerful and fast Z80 Basic interpreter with EDIT, RENUMBER, TRACE, PRINT USING, assembly language subroutine CALL, LOADGO for "chaining", COPY to move text, EXCHANGE, KILL, LINE INPUT, error intercept, sequential file handling in both ASCII and binary formats, and much, much more. It runs in a little over 12 K. An excellent choice for games since the precision was limited to 7 digits in order to make it one of the fastest around. $49.95/$15.

### BASIC II

Basic I but with 12 digit precision to make its power available to the business world with only a slight sacrifice in speed. Still runs faster than most other Basics (even those with much less precision). $99.95/$15.

### BUSINESS BASIC

The most powerful Basic for business applications. It adds to Basic II with random or sequential disk files in either fixed or variable record lengths, simultaneous access to multiple disk files, PRIVACY command to prohibit user access to source code, global editing, added math functions, and disk file maintenance capability without leaving Basic (list, rename, or delete). $179.95/$25.

### ZEDIT

A character oriented text editor with 26 commands and "macro" capability for stringing multiple commands together. Included are a complete array of character move, add, delete, and display function. $49.95./$15.

### ZTEL

Z80 Text Editing Language - Not just a text editor. Actually a language which allows you to edit text and also write, save, and recall programs which manipulate text. Commands include conditional branching, subroutine calls, iteration, block move, expression evaluation, and much more. Contains 36 value registers and 10 text registers. Be creative! Manipulate text with commands you write using Ztel. $79.95/$25.

### TOP

A Z80 Text Output Processor which will do text formatting for manuals, documents, and other word processing jobs. Works with any text editor. Does justification, page numbering and headings, spacing, centering, and much more! $79.95/$25.

### MACRO I

A macro assembler which will generate relocateable or absolute code for the 8080 or Z80 using standard Intel mnemonics plus TDL/Z80 extensions. Functions include 14 conditionals, 16 listing controls, 54 pseudo-ops, 11 arithmetic/logical operations, local and global symbols, chaining files, linking capability with optional linker, and recursive/reiterative macros. This assembler is so powerful you'll think it is doing all the work for you. It actually makes assembly language programming much less of an effort and more creative. $79.95/$20.

### MACRO II

Expands upon Macro I's linking capability (which is useful but somewhat limited) thereby being able to take full advantage of the optional Linker. Also a time and date function has been added and the listing capability improved. $99.95/$25.

### LINKER

How many times have you written the same subroutine in each new program? Top notch professional programmers compile a library of these subroutines and use a Linker to tie them together at assembly time. Development time is thus drastically reduced and becomes comparable to writing in a high level language but with all the speed of assembly language. So, get the new CDL Linker and start writing programs in a fraction of the time it took before. Linker is compatible with Macro I & II as well as TDL/Xitan assemblers version 2.0 or later. $79.95/$20.

### DEBUG I

Many programmers give up on writing in assembly language even though they know their programs would be faster and more powerful. To them assembly language seems difficult to understand and follow, as well as being a nightmare to debug. Well, not with proper tools like Debug I. With Debug I you can easily follow the flow of any Z80 or 8080 program. Trace the program one step at a time or 10 steps or whatever you like. At each step you will be able to see the instruction executed and what it did. If desired, modifications can then be made before continuing. It's all under your control. You can even skip displaying a subroutine call and up to seven breakpoints can be set during execution. Use of Debug I can pay for itself many times over by saving you valuable debugging time. $79.95/$20.

### DEBUG II

This is an expanded debugger which has all of the features of Debug I plus many more. You can "trap" (i.e. trace a program until a set of register, flag, and/or memory conditions occur). Also, instructions may be entered and executed immediately. This makes it easy to learn new instructions by examining registers/memory before and after. And a RADIX function allows changing between ASCII, binary, decimal, hex, octal, signed decimal, or split octal. All these features and more add up to give you a very powerful development tool. Both Debug I and II must run on a Z80 but will debug both Z80 and 8080 code. $99.95/$20.

### ZAPPLE

A Z80 executive and debug monitor. Capable of search, ASCII put and display, read and write to I/O ports, hex math, breakpoint, execute, move, fill, display, read and write in Intel or binary format tape, and more! on disk

### APPLE

8080 version of Zapple

### NEW! TPM now available for TRS-80 Model II!

### TPM*

A NEW Z80 disk operation system! This is not CP/M*. It's better! You can still run any program which runs with CP/M* but unlike CP/M* this operating system was written specifically for the Z80* and takes full advantage of its extra powerful instruction set. In other words its not warmed over 8080 code! Available for TRS-80* (Model I or II). Tarbell, Xitan DDDC, SD Sales "VERSA-FLOPPY", North Star (SD&DD), and Digital (Micro) Systems. $79.95/$25.

### SYSTEM MONITOR BOARD (SMB II)

A complete I/O board for S-100 systems. 2 serial ports, 2 parallel ports, 1200/2400 baud cassette tape interface, sockets for 2K of RAM, 3-2708/2716 EPROM's or ROM, jump on reset circuitry. Bare board $49.95/$20.

### ROM FOR SMB II

2KX8 masked ROM of Zapple monitor. Includes source listing $34.95/$15.

### PAYROLL (source code only)

The Osborne package. Requires C Basic 2.
5" disks $124.95 (manual not included)
8" disks $ 99.95 (manual not included)
Manual $20.00

### ACCOUNTS PAYABLE/RECEIVABLE (source code only)

By Osborne, Requires C Basic 2
5" disks $124.95 (manual not included)
8" $99.95 (manual not included)
Manual $20.00

### GENERAL LEDGER (source code only)

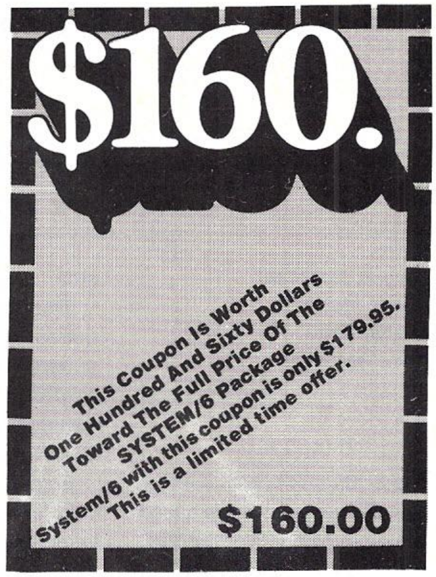
By Osborne. Requires C Basic 2
5" disks $99.95 (manual not included)
8" disks $99.95 (manual not included)
Manual $20.00

### C BASIC 2

Required for Osborne software. $99.95/$20.

### SYSTEM/6

TPM with utilities, Basic I interpreter, Basic E compiler, Macro I assembler, Debug I debugger, and ZEDIT text editor.
Above purchased separately costs $339.75
Special introductory offer: Only $179.75 with coupon!!




### ORDERING INFORMATION

Visa, Master Charge and C.O.D. O.K. To order call or write with the following information.

1. Name of Product (e.g. Macro I)
2. Media (e.g. 8" CP/M)
3. Price and method of payment (e.g. C.O.D.) include credit card info. if applicable.
4. Name, Address and Phone number.
5. For TPM orders only: Indicate if for TRS 80, Tarbell, Xitan DDDC, SD Sales (5¼" or 8"). ICOM (5¼" or 8"), North Star (single or double density) or Digital (Micro) Systems.
6. N.J. residents add 5% sales tax.

Manual cost applicable against price of subsequent software purchase in any item except for the Osborne software.

**For information and tech queries call**
**609-599-2146**

**For phone orders ONLY call toll free**
**1-800-327-9191**
**Ext. 676**
(Except Florida)

### OEMS

Many CDL products are available for licensing to OEMs. Write to Carl Galletti with your requirements.

* Z80 is a trademark of Zilog
* TRS-80 is a trademark for Radio Shack
* TPM is a trademark of Computer Design Labs. It is not CP/M*
* CP/M is a trademark of Digital Research

Prices and specifications subject to change without notice.

**DEALER INQUIRIES INVITED.**

COMPUTER DESIGN LABS ✓18

342 Columbus Avenue
Trenton, N.J. 08629

# Tracking Down The Bus

*Answers the question of why some boards won't work with your S-100 system.*

*Richard A. Rodman*
*c/o Envo, Inc.*
*800 Follin Lane*
*Vienna, VA 22180*

The S-100 bus has one intrinsic characteristic: it's based on the timing signals of the 8080 microprocessor. Thus, to adapt bus, the signals generated by the 8080 have to be simulated by the Z-80 CPU circuit design.

This is not easy. The Z-80 requires only a single clock and generates much more elegant control signals, while the 8080 requires a two-phase clock with 12-volt swings.

Engineers have found that the simplest way to simulate the two-phase clock on the bus is to use the 8224 clock generator chip. While the engineers at Cromemco did not follow this route, Ithaca Intersystems did; theirs is the most popular Z-80 CPU board in use.

Many boards, however, do not work with this CPU, even in its latest version. The problem is related to the S-100 interface logic, and the phase of clock 2 and clock 1 to the PSYNC signal. The 8080 CPU generates PSYNC on the leading edge of clock 2. Clock 2 makes one high-to-low transition during PSYNC, and clock 1 makes a low-to-high transition.

Peripheral boards use these relative characteristics in a number of ways.

Dynamic memory boards use the high-to-low transition of clock 2 during PSYNC to latch in the address, which is stable at the time. Examples of this type are the Processor Technology 16KRA and 32KRA memory boards.

Other boards, such as the Solid State Music VB1 and many static memory boards, use the low-to-high transition of clock 1 for the same purpose.

Finally, some boards will use the falling edge of clock 2 and the rising edge of PSYNC to toggle circuitry for the generation of wait states. These include the Screensplitter and Alpha Micro AM-200 floppy-disk controller.

Why will some of these boards work as is, others poorly and most not at all?

Fig. 1 shows the circuitry used by Ithaca Intersystems to generate PSYNC. The Ø2TTL output of the 8224 is used to operate the Z-80 clock, whereas the standard outputs Ø1 and Ø2, powered from a 5-volt supply, are buffered onto the bus. Note also that the Z-80 output/MREQ (not memory request) drives a 74121 monostable to generate the PSYNC pulse on its falling edge.

A timing diagram of these signals is given in Fig. 2. Since the Z-80 outputs /MREQ low on the falling edge of the clock applied to it, that PSYNC will make its low-to-high transition on the falling edge of Ø2, rather than the rising edge. This means that while Ø1 will make its required transition, Ø2 will not make a high-to-low transition during PSYNC.

Since the Z-80 doesn't use Ø1, simply invert Ø2TTL before it gets to the Z-80, so that /MREQ will make PSYNC occur at the right time.

I wrote Ithaca about this problem, but they weren't interested, so you'll have to modify your own board. Refer to Fig. 3 for details. All mods are done on the solder side of the board.

1. Remove R6 (330 Ω orange-orange-brown) carefully and save it.
2. Cut the third trace from the top on the solder side, right near U12 where it makes an angle (see Fig. 3).
3. With U12 at the top, connect the side on the right to U12, pin 5.
4. Connect the left side (going to the CPU) to pin 6. The wires should criss-cross. As with all PCB mods, use #30 Kynar wire-wrapping wire or equivalent.
5. Connect R6 between pin 14 and pin 6 of U12.

This completes the modification, and all of the boards mentioned above should operate correctly. ■

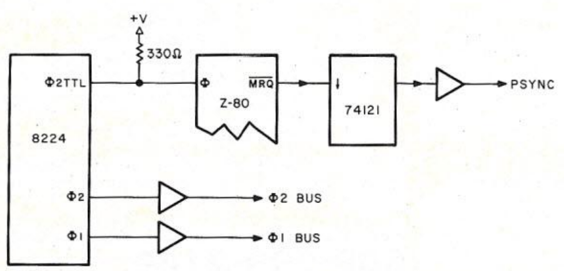



*Fig. 1.*

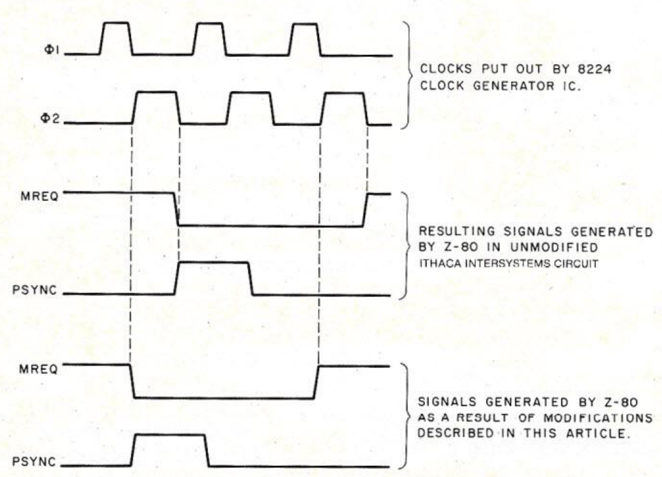



*Fig. 2.*

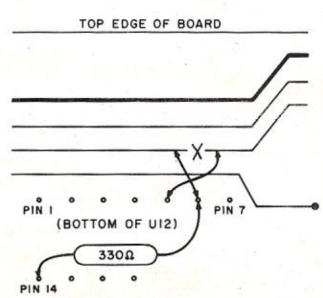



*Fig. 3. Bottom of board modification.*

# M-n-M's

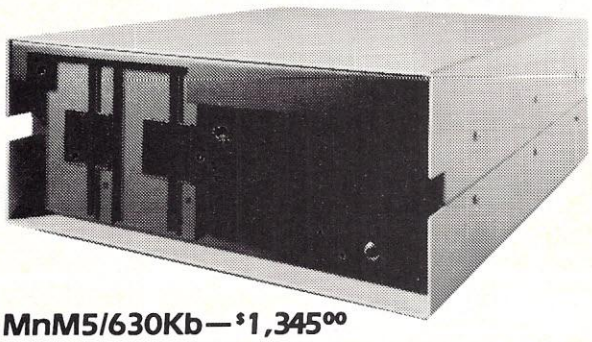
## S-100 MAINFRAMES-N-MEMORY

**High quality S-100 mainframes with integrated dual disk drives, need only your standard S-100 boards (or ours) to complete the system.**

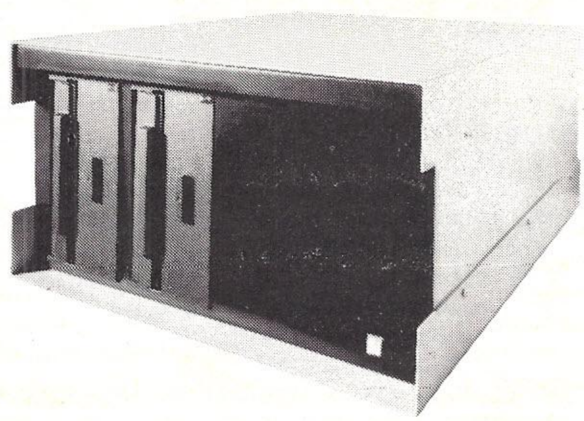
Standard features include reset-glo-button, keyed on/off power switch that also powers-on dual outlets on the rear panel for peripheral devices. 'Quasi-coaxial' S-100 Motherboard* with adaptive termination assures you of 'silent' operation. Heavy-duty linear power supply supports both the Motherboard and disk drives.

*All have 10 slot Motherboard/card guides
*Optional rack mount version with only 5 slots—but is available in 2-cabinet version with 15 slots and horizontal dual drives in second cabinet.

Delivery—2 to 4 Weeks

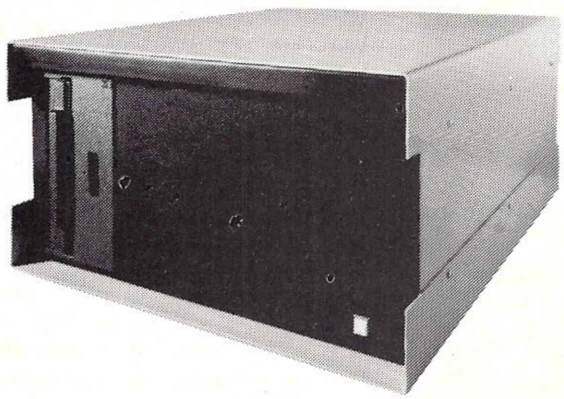


**MnM5/630Kb—$1,345.00**
Dual Micropolis 1015-MOD II's



**MnM8/1Mb—$1,695.00**
Dual Remex RFD—2,000's

**MnM8/2Mb—$2,145.00**
Dual Remex RFD—4,000's



**MnM-HD8/7 + 1Mb—$4,695.00**
**MnM-HD8/21 + 1Mb—$5,695.00**
**MnM-HD8/35 + 1Mb—$6,595.00**
8" Micropolis Hard Disk + 1 RFD-4000 Floppy Disk

**OEM SYSTEMS & COMPONENTS**

INTELLIGENT BUSINESS MACHINES CORPORATION
1612 E. Shields • Fresno, CA 93704 • (209) 222 • 3302

✓103

# Dial-up Directory

*Meet the father of Forum-80, Bill Abney.*

*Frank J. Derfler, Jr.*
*PO Box 691*
*Herndon, VA 22070*

If you've read the Dial-up Directory listings, you've seen the title Forum-80. In previous articles I've examined Computer Bulletin Board Systems Chicago-style, Apple Bulletin Board Systems by Bill Blue and two North Star systems. For this issué, I visited the creator of the Forum-80, Bill Abney, to find out more about making Radio Shack's TRS-80 into an electronic bulletin board and program transfer system.

**Bill**: "I'm only one deep, Frank. I receive calls from my Forum-80 operators around the country and constantly try to help them with problems. I have almost no time to answer other inquiries. My stack of letters right now is about four inches deep."

**Microcomputing**: "You used a phrase there, Bill. You said, 'my operators.' It has been said that you have a very strong proprietary feeling about Forum-80 systems, despite the fact that each one is run by other private individuals."

**Bill**: "I refer to the Forum-80 network. It is not a network of stations that is permanently interconnected, but the interconnection is there at several levels. The users check into different systems and carry information between them. The system operators also swap messages between the systems.

"I feel it is very important to keep compatibility and uniformity between the Forum-80 operations. Some operators would like to make changes in their systems to provide unique services, but they can't do it with the existing software. The version 2.1 software has been expanded about as far as it can go while still maintaining uniformity. There have been other problems too."

**Microcomputing**: "Like what?"

**Bill**: "Software sharks for one. Bandits who make very small changes in my software and sell it themselves. I stopped putting out any more 2.1 software in December 1979 and fired up a version 3.0 in February 1980. The new version will give the Forum operators the flexibility to provide all the special features they like while maintaining uniformity for the user. Version 3.0 will be provided to the Forum operators, but they will not own it. It will be licensed like other commercial software. New licenses will cost about $150."

**Microcomputing**: "How will you allow more versatility in the program?"

**Bill**: "Version 3.0 will allow branching into special-purpose subroutines, but the system will initially come up looking like any other Forum-80 to the users. Users can then treat the system as a standard Forum-80 or move into special uses."

| | | | |
|---|---|---|---|
| Forum Augusta, GA | (3.0) | 803-279-5392 | |
| Forum Boston, MA | (3.0) | 617-431-1699 | |
| Forum Chicago, IL | (2.5) | 312-782-8180 | |
| Forum Cleveland, OH | (3.0) | 216-486-4176 | |
| Forum Dallas, TX | (3.0) | 214-288-4859 | |
| Forum Denver, CO | (3.0) | 303-789-0936 | |
| Forum Fairfax, VA | (3.0) | 703-978-7561 | (Genealogy) |
| Forum Ft. Lauderdale, FL | (3.0) | 305-772-4444 | |
| Forum Kansas City, MO | (3.0) | 816-861-7040 | (H.Q. SYS) |
| Forum Kansas City, MO | (3.0) | 816-931-9316 | (Commodities) |
| Forum Las Vegas, NV | (3.0) | 702-873-9491 | |
| Forum Memphis, TN | (3.0) | 901-276-8196 | |
| Forum Memphis, TN | (3.0) | 901-362-2222 | (Hobbyist) |
| Forum Mt. Clemens, MI | (3.0) | 313-465-9531 | (Medical) |
| Forum San Francisco, CA | (3.0) | 415-348-2139 | |
| Forum Tampa, FL | (2.1) | 813-223-7688 | |
| Forum Union, NJ | (3.0) | 201-688-7117 | |
| Forum Ventura County, CA | (2.1) | 805-484-9904 | |
| Forum Wichita, KS | (3.0) | 316-746-2078 | |
| Forum Wichita Falls, TX | (2.1) | 817-855-3916 | |

*In honor of our interview with Forum-80 founder, Bill Abney, I will limit our system list this month to Forum systems. Bill has provided this current (as of publication deadline) list of active Forum-80 systems. The number in parentheses refers to the version of software in use. Version 3.0 has the most complete set of features. Some of these numbers may be redundant to previous lists. Some of the most interesting special-user systems are running with the Forum-80 software.*

*Bill Abney, the Forum-80 founding father. Bill provides a lot of support for his Forum-80 operators, but wishes he had time to do more.*

**Microcomputing**: "You mean uses like family history and medical applications?"

**Bill**: "Yes. I have heard of a special-interest system for photographers too. Some operators may want to provide special services, graphics, programs to run, user codes and many other things. These submenus can be used by the people they are intended for, while other, less sophisticated, users see the basic Forum-80 menu come up in default. I would expect the operators to contribute submenu software to the network pool so it can be shared where appropriate."

**Microcomputing**: "Aside from the software, what does a person need to set up a Forum-80?"

**Bill**: "They need a TRS-80 with 48K of memory. This means they need the expansion interface. It has to have the RS-232 card installed. They need three disk drives for version 3.0. With the older version the third disk was optional, but it is needed in this version. They also need an automatic answer modem. This will give the full Forum capability."

**Microcomputing**: "That looks like a $3500 system at retail prices. What kind of modem do you recommend?"

**Bill:** "I think the U.S. Robotics auto answer modem is a tremendous piece of equipment. It is well designed and interfaces with the TRS-80 very well."

**Microcomputing**: "The Forum-80 in Kansas City (816-861-7040) is referred to as the 'headquarters.' Is it a very active system?"

**Bill**: "This system is more for demonstration than anything else. It is the only computer I own, and I use it for all the development work, so it is on the phone as a forum at odd hours. However, during the online hours, the Forum-80 Headquarters System serves many regular and long-distance users."

**Microcomputing**: "How did you arrive at the Forum-80 name?"

**Bill**: "I felt that the term Bulletin Board implied merely a place to post notices. A forum is more a place of public discussion. In this view, I hoped the Forum would become a place where the microcomputer user could discuss his pursuits and exchange intelligence with others. I feel the educational value is immense, both for the beginners and the experienced users. The 80 in the name, of course, came from the TRS-80, but with the 3.0 system we are attempting to achieve compatibility with other machines and prefer to look at it as the 'Forum of the 80s.' "

**Microcomputing**: "What else would you like our readers to know, Bill?"

**Bill**: "That's easy! Please tell them to use and read the help commands available on every system. If they use trial and error learning to find all of the Forum's features, they will be frustrated, will probably miss something and will keep the system busy for much longer than it needs to be. If they send a self-addressed legal-size envelope with two first-class stamps attached, I will send them a Forum-80 user's guide. I would much rather pay to print up this guide than have people tie up systems. It is nothing fancy, but it is useful. I hope they read it."

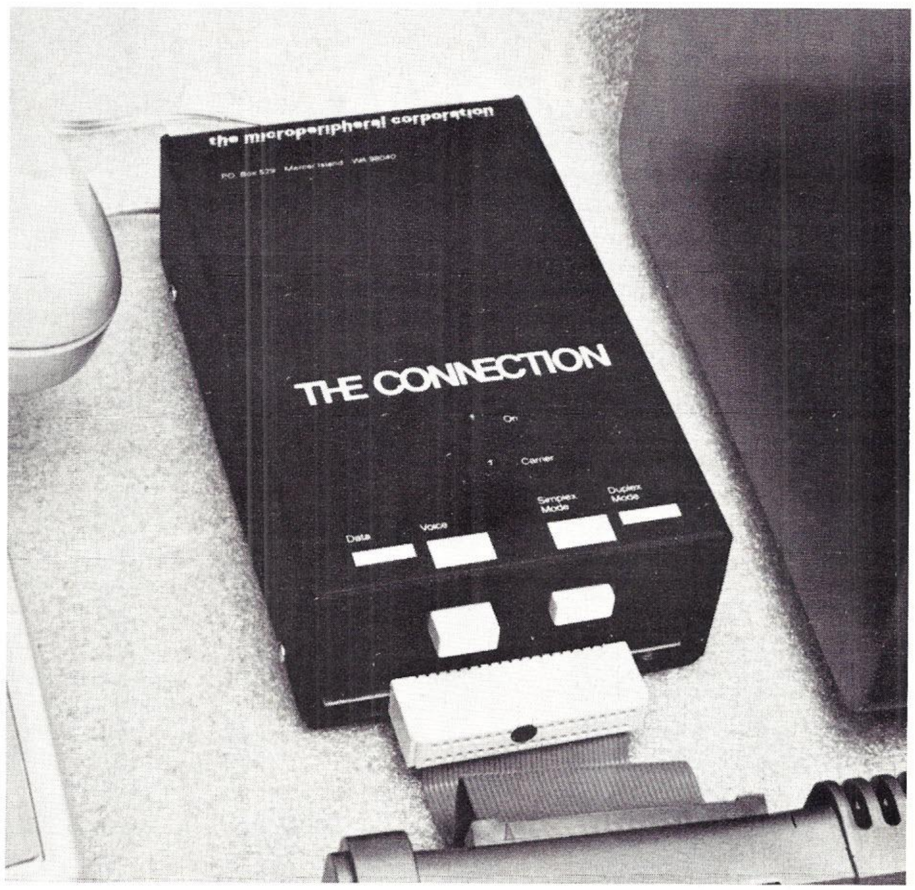
Send your large double-stamped self-addressed envelopes to: Forum-80 Headquarters, 7600 E. 48th Terrace, Kansas City, MO 64129.

### The Micro-Connection

Here are two names for you: The Peripheral People and Don Stoner. The first name was new to me, but the second has been familiar for a long time. Don Stoner is better known to the amateur radio world as W6TNS. He has been involved in radio equipment design and manufacturing at several different levels. The Peripheral People is the name of Don's company in Washington state.

These names are interesting, particularly to TRS-80 users, because of their new product called the Micro-Connection. Basically a modem, this device connects directly to the phone line so you don't need an acoustic coupler. (Acoustic couplers and telephone handsets sometimes cause distortion, and can be the entry point for disruptive noise.)

This device will interface with all TRS-80 models I, II and III, as well as the TRS-80 Color Computer; any memory size; any type of BASIC; and with or without an expansion interface. Previously, a computerist needed




*The Micro-Connection modem in operation. It is a small, convenient device with great capabilities.*

the expansion interface and had to install the RS-232 card to use a modem for telecommunications. This called for a $400 investment before you even got to the modem price.

The Micro-Connection is selling for $249. It plugs into the TRS-80 bus connector, or can plug directly into the screen printer port of the expansion module.

The Micro-Connection also has a female RS-232 output connector on the modem. Thus, a serial printer can print along with the modem and provide hard copy of everything on the screen, without fancy software. This provides a port that is separate from the regular TRS-80 RS-232 card. You can operate the Micro-Connection as just an interface between the TRS-80 and any RS-232 device—it does not have to be operating as a modem to give RS-232 output. This means you can run a series printer without the expansion interface and RS-232 card. Do you understand why they call it the Micro-Connection?

Finally, since Don is an active amateur radio operator, the literature for the Micro-Connection points out that you can use this device with an amateur radio station to transmit and receive ASCII. This can be done with any modem, of course, but the configuration of the Micro-Connection makes it easy to get at the audio input and audio output lines for interconnection to the radios. Since the FCC made ASCII transmission legal in March, this has become an very important consideration for modem users and manufacturers.

For more information on the Micro-Connection, contact The Peripheral People at PO Box 524, Mercer Island, WA 98040. They also run their own bulletin board system at 206-723-DATA.

#### Random Noise

Some users of Novation CAT and other acoustically coupled modems have written to say that they have some intermittent distortion problems that seem to involve the telephone handset. The cures include everything from putting cotton in the earpiece cup to using strange chants and incantations.

These cures sound like the medieval treatment for the plague, but there is a little science behind them. Part of the science involves doing something—anything—to unpack the carbon granules in the telephone mouthpiece. The old-time ham radio operators used to broadcast with a pencil in one hand and tap the microphone periodically to unpack the carbon.

Got a transmission problem with an acoustically coupled modem? Made sure the phone is tightly seated and the environment is free of outside noise and bang the phone mouthpiece against your palm once or twice. It might help.

Also, the carbon microphones create a second harmonic feedback through the handset back to the earpiece, thereby creating distortion in the modem input side. That may explain why cotton around the inside of the earpiece sometimes helps. It can dampen the second harmonic vibration. Just don't cover up the sound holes or cut down on the level of the tones you want.

Novation is coming out with a dynamic element to replace the carbon one in your telephone if it becomes a persistent problem. This element will do away with both carbon packing and second harmonic resonance.

This is another argument in favor of direct-coupled modems. They aren't as portable, but they may provide more reliable operation.

#### Data Transfer

Send me your data! If you have comments, questions or information about computer bulletin board services or other aspects of data transmission, let me know. Use the address at the beginning of the article or send it to TCB967 on The Source. Send a stamped envelope if you want a response to paper mail. ■

✓ Reader Service index—page 241

# Reduce Search Time With an Index

*The author shows how you can breeze through file searches with this Heath H8 program.*

*LeRoy E. Kolderup*
*1497 Sugartown Road*
*Paoli, PA 19301*

Microprocessors are frequently used in business applications to search long files for a specific record. Such a search is usually made by moving the file into main memory and using a BASIC routine of the type shown in Listing 1, where a name (N$) is used as a key to locate a record (M$[K]). A print of this record supplies the desired information contained in the remainder of the record, which might include address, date of last order and so on.

The limitations imposed by BASIC and by sequential files make a search through several hundred records a time-consuming task that would try anyone's patience. But the time can be significantly reduced by using an index.

A file can be indexed in many ways, but some methods are elaborate and unsuitable for use on a small system with limited memory. The indexing system described here uses less than 100 bytes of memory, can be easily generated and will reduce search time by a factor of from ten to 20.

In a business application, the principle key with which a search is made might be the name of an individual or company. The records can be arranged in alphabetic order of the principle key. The index system that will be described supplies the record number of the first record that begins with each letter of the alphabet. This information is then used to limit the area of search to a specific portion of the file.

**Generating the Index System**

The first step in generating this index is to sort the file, if it is not already in alphabetic order. This step need only be repeated when additional records are added to a file. For files of any significant length, the sort technique should be more advanced than the overused bubble sort to avoid excessive sorting times. Such techniques as the Shell-Metzner sort or Hoare's Quicksort have been described in detail and will not be repeated here (see "Quicksort," *Microcomputing*, April 1979, p. 96).

The second step generates an index that lists the record number of the first record that starts each alphabetic character group. If the file contains records that begin with the letter A, its index value will, of course, be 1. If the first record that has a key beginning with the letter G is record number 145, then the index for G will be 145. If there is no record beginning with a particular alphabetic character, its index is made equal to the number of the last record in the file.

```
>LIST EXAMPLE.BAS
00120 INPUT "NAME? ";N$
00130 IF N$="STOP" THEN 190
00140 FOR K=1 TO 400
00150 IF LEFT$(M$(K),LEN(N$))=N$ THEN PRINT M$(K):GOTO 120
00160 NEXT K
00170 PRINT "NO MATCH FOUND"
00180 GOTO 120
00190 END
>
```

*Listing 1. Typical file search routine.*

A program to generate this index is given in Listing 2. I presume that the file to be indexed is formatted with the keyword at the beginning of each record, has been sorted alphabetically and resides on disk. Lines 10 through 80 constitute the input routine to load the file into memory. This program was prepared on a Heath H8, and some modifications may be required for other disk systems or for a tape system.

The last record in the file is a special record that contains two numeric zeros in character format (00) as the two leading characters. Note that detection of this pair of zeros in line 50 serves as a convenient end-of-file marker. If the file has not been previously indexed, then this character pair should be appended as the final record before using the index generator.

Lines 120 through 190 generate the index, I(A) for A = 1 to 26. Line 140 initially compares the ASCII code of the first character of the record with the code for the character A. If they do not compare, no records start with A, and the index value assigned to I(1) is the value of the final record. The next character in the alphabet is then selected and the process is repeated. If the comparison is true, then I(1) is set equal to one in line 150.

Line 160 selects the next record and checks for the end-of-file in line 170. Line 180 compares the first character of the record with that of the previous record and continues selecting the next record until a line beginning with a different character is reached. The program then goes to the next alphabetic character and repeats until all characters of the alphabet have been indexed by assigning the number of the record that first begins with each letter of the alphabet to the index I(A).

Lines 210 through 240 pack this index into a single string that is structured to require a minimum of file space and can be

```
>LIST INDEXGEN.BAS
00005 REM FILE INPUT ROUTINE
00010 DIM M$(400),I(26)
00020 OPEN "TESTFILE" FOR READ AS FILE #1
00030 FOR K=1 TO 400
00040 LINE INPUT #1,;M$(K)
00050 IF LEFT$(M$(K),2)="00" THEN 70
00060 NEXT K
00070 CLOSE #1
00080 Z=K-1
00090 REM END OF FILE INPUT
00110 REM BEGIN INDEX GENERATION
00120 K=1
00130 FOR A=1 TO 26
00135 REM CHECK FOR MATCH OF 1ST CHARACTER OF RECORD AND ALPHA CHARACTER
00140 IF ASC(M$(K))<>A+64 THEN I(A)=Z:GOTO 190
00145 REM ASSIGN RECORD NO. AS INDEX VALUE
00150 I(A)=K
00155 REM SELECT NEXT RECORD, CHECK FOR END OF FILE
00160 K=K+1
00170 IF LEFT$(M$(K),2)="00" THEN 190
00175 REM IF 1ST CHARACTER OF RECORD MATCHES THAT OF PRECEEDING RECORD
00177 REM          THEN NEXT RECORD
00180 IF ASC(M$(K))=ASC(M$(K-1)) THEN 160
00190 NEXT A
00200 REM GENERATE INDEX STRING AND APPEND TO FILE
00210 I$="00"
00220 FOR A=1 TO 26
00230 I$=I$+LEFT$(RIGHT$("  "+STR$(I(A)),4),3)
00240 NEXT A
00250 M$(Z+1)=I$
00260 REM END OF INDEX GENERATION
00290 REM WRITE FILE WITH INDEX APPENDED BACK TO DISK
00300 OPEN "TESTFILE" FOR WRITE AS FILE #1
00310 FOR K=1 TO Z+1
00320 PRINT #1,M$(K)
00330 NEXT K
00340 CLOSE #1
00350 END
>
```

*Listing 2. Program to generate a file index.*

```
*PRINT I$
00   1 16 58 94106118145178387211217238259280289292387310328358387387367387382387
*
```

*Listing 3. Format of index string.*

easily decoded in the search routine. This string begins with the character pair 00 and continues with three digits allocated for each index entry. Line 230 converts the index value for each I(A) to character format, strips the trailing zero and either adds or strips leading zeros to make all entries have a length of three characters. (A file with more than 999 records will require this routine to be modified to allow for four characters.) These values are then strung together to form the index string.

Listing 3 illustrates the format of the index string for a typical file. In this example, the first record in the file begins with A, record 16 is the first record beginning with B, record 58 with C and so on. No records begin with the letters I, Q, U, V, X and Z; therefore, these letters are indexed to 387, the last record in the file. Line 250 appends this index string to the end of the file. Lines 300 through 340 write the file with the appended index back to disk.

**Searching the Index**

The indexed file is now ready for a rapid search. Changes to a record (other than the first character of the keyword) or deletion of a record can be made without requiring the generation of a new index.

A typical program for searching the indexed file is shown in Listing 4. Lines 10 through 70 contain the input routine to load the file together with the index string into memory. Lines 80 through 110 decode the index string into 26 index values for I(A). These values are then used in line 140 of the search routine to confine the search to those records that begin with the same character as the key. If no records that start with this character are present, the search is indexed to the end of the file and the "NO MATCH FOUND" message is immediately displayed.

The program was tested using a file containing 387 records. The index generation process (lines 120 through 250, Listing 2) was completed in 90 seconds after the file was loaded into memory. (File load and dump time will vary with record lengths and distribution of a file on the disk sectors.)

The file was then searched for several records scattered throughout, using the last name as a key and printing out the complete record as the response. I used the program contained in Listing 4. The response times varied from one to six seconds, with an average of 2.6 seconds.

I repeated the test without the benefit of an index by changing line 140 to read FOR K = 1 TO 400 and deleting lines 80 through 110 from the program in Listing 4. The file was searched for the same records as in the previous test. In this case, the response times varied from three to 66 seconds. As you would expect, the time for any given record was roughly proportional to the record number. The average time of 38 seconds was about 15 times longer than that required with an index.

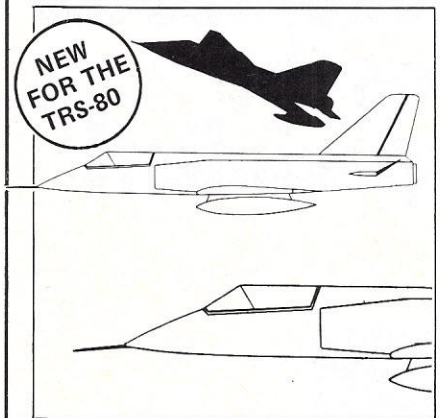
The above indexing scheme is one of several possible methods of using an index. This one requires little additional memory, has a simple index generation program and requires only a few additional lines to the search routine. It is directly applicable to almost any database file and can be adapted with minor modifications to handle a case where the keyword is located at other than the beginning of a record. It can be used to quickly locate records for both display and update purposes.

Application of this technique to a file with as few as 100 records will pay off in reduced time to locate specific records for display, update or other processing. For larger files, the time saved is significant, and efficient data processing demands the use of an index. ■

```
>LIST QSEARCH.BAS
00005 REM FILE INPUT ROUTINE
00010 DIM M$(400),I(26)
00020 OPEN "TESTFILE" FOR READ AS FILE #1
00030 FOR K=1 TO 400
00040 LINE INPUT #1,;M$(K)
00050 IF LEFT$(M$(K),2)="00" THEN 70
00060 NEXT K
00070 CLOSE #1
00075 REM DECODE INDEX STRING
00080 I$=M$(K)
00090 FOR A=1 TO 26
00100 I(A)=VAL(MID$(I$,3*A,3))
00110 NEXT A
00115 REM FILE SEARCH ROUTINE
00120 INPUT "NAME? ";N$
00130 IF N$="STOP" THEN 190
00140 FOR K=I(ASC(N$)-64) TO I(ASC(N$)-63)
00150 IF LEFT$(M$(K),LEN(N$))=N$ THEN PRINT M$(K):GOTO 120
00160 NEXT K
00170 PRINT "NO MATCH FOUND"
00180 GOTO 120
00190 END
>
```

*Listing 4. Search program for an indexed file.*

# Great Simulations




## Jet Fighter Pilot

The Jet Fighter Pilot package takes you as close to real combat flying as possible. . . without pulling G's.

In this brilliantly realistic simulation, you become the pilot of a high performance, twin turbo-jet fighter. Total control of the aircraft is yours.

At the start of your mission, you'll go through an entire engine start procedure before your flight (provided your ground maintenance is up to par). Your takeoff will be from either the deck of an aircraft carrier (via a steam catapult) or from an airfield.

All controls respond the same as they would on a real jet fighter. You'll have to constantly monitor your display and make adjustments to your throttle, flaps, rudder and air spoilers. You decide when to retract flaps, landing gear and release the auxiliary fuel drop-tanks.

Your on-board navigational computer will direct you to your selected airport. The Glideslope/Localizer information will aid you in approaching and landing on an aircraft carrier deck or airfield.

The Weapons Control Computer will arm your missiles, provide you with the range and bearing to a target, and tell you when to attack. And, if things should get a little too hot, you have an ejection seat command for egress.

For a carrier-based landing, you'll have to deploy your tail hook. For a land-based landing, you'll need reverse thrust and your drag chute.

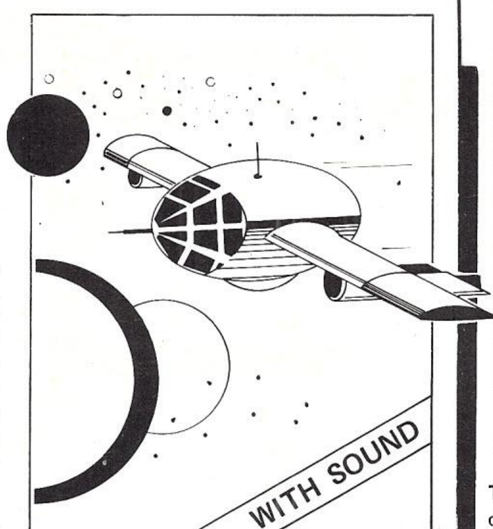
After you've flown a few missions with the Jet Fighter Pilot package, you'll *know* you've earned your wings.

**Order No. 0159R $14.95**

NEW FOR THE TRS-80

## Cosmic Patrol

WARNING: PLAYERS OF THIS GAME SHOULD BE PREPARED FOR A STATE OF REALISM HITHERTO UNAVAILABLE ON THE TRS-80




The Cosmic Patrol program puts you in the command chair of a small interstellar patrol craft. Your mission is to defeat Terran space and prey on the Quelon supply ships which carry essential parts and lubrificants for that implacably hostile robotic force. The drone freighters are fairly easy pickings for the accomplished starship pilot, but beware of the I-Fighter escorts. They're armed, fast and piloted by intelligent robots linked to battle computers. They *never* miss.

The Cosmic Patrol program is not just another search and destroy game. With its fast, real-time action, impressive sound option and superb graphics, this machine-language program is the best of its genre.

Don't keep putting quarter after quarter into arcade games or spending big bucks for video games cartridges. Get Cosmic Patrol from Instant Software—and get the best for less!

**Order No. 0223R $14.95**


## Battleground

It is late 1944, and the Allied Forces are sweeping toward Berlin. As General in command of your sector, you study the map. At your command, are tanks, planes, artillery, infantry, engineers and vehicles—an awesome array of fighting men and the machines of war. From Intelligence reports you know that the enemy General is a shrewd tactician, not to be under-estimated. It will take planning and strategy to outwit this wily old campaigner.

The battle map of your sector will fill with markers, each showing the deployment of your forces. You and another player will slip into the roles of opposing German and American commanders as yet another battle unfolds.

Battleground allows you to experience the awesome responsibility of a battle-area command. It will be up to you to deploy your tanks, planes, vehicles, weapons and men. On your shoulders rests the decision, whether to call for direct artillery gunfire, or to order your planes into the air. You will constantly be watching for an enemy air-drop, always carefully maneuvering your forces.

The stark reality of World War II comes alive in Battleground.

**Order No. 0141R $9.95**

**TO ORDER:** Look for these programs at the dealer nearest you (see list of dealers on page 199). If your store doesn't stock Instant Software send your order with payment to:

Instant Software
Order Dept.
Peterborough, N.H. 03458

(Add $1.00 for handling) or call toll-free 1-800-258-5473 (VISA, MC and AMEX accepted).

# Instant Software™

Prices subject to change without notice.

PETERBOROUGH, N.H. 03458
603-924-7296

# Video HARDCOPY For CP/M

*These machine-language programs give you printing power at the touch of a key.*


*Glenn Stok*
*PO Box 501*
*Woodside NY 11377*


How often have you wanted to keep a screen of information permanently? This desire has occurred to me many times. So I decided to add a hard-copy feature to my CP/M system.

In this article I will explain the method, show you the source listing and explain how to incorporate it into your CP/M system or your monitor. The only hardware requirement is that you have a memory-mapped video display and a printer. If you don't have CP/M, you can still use my HARDCOPY routine by making it part of your monitor (see Listing 1).

If you have a Z-80 CPU, you may find Listing 2 helpful. It is the same HARDCOPY routine written in Z-80 code. I have restricted this to relative addressing so it is relocatable. All you have to do is patch the machine code of Listing 2 anywhere in your monitor and patch branches to and from it for I/O. The CALL to your PRINT routine has to be patched in three places. Make sure that you correct the address in the CALL CRLF instructions (the Z-80 does not have relative addressable CALL instructions). Listing 1 is entirely in 8080 code, so it will run on either CPU.

```
CC00 =          SCRN    EQU     0CC00H  ;VIDEO SCREEN ADDRESS
0100                    ORG 100H
0100 C5                 PUSH B  ;SAVE REGS
0101 E5                 PUSH H
0102 2100CC             LXI     H,SCRN  ;SCREEN ADDRESS
0105 7E         VP1:    MOV     A,M     ;LOOK FOR FIRST
0106 FE20               CPI     ' '     ;NON-BLANK LINE.
0108 C20F01             JNZ     VP2
010B 23                 INX     H
010C C30501             JMP     VP1
010F 7D         VP2:    MOV     A,L
0110 E6C0               ANI     11000000B       ;SET TO LINE BEGINNING
0112 6F                 MOV     L,A
0113 7D         VP3:    MOV     A,L             ;PRINT REST OF SCREEN
0114 E63F               ANI     00111111B       ;CHECK END OF LINE
0116 C22F01             JNZ     VP4
0119 CD4101             CALL    CRLF
011C E5         VP5:    PUSH    H               ;CHECK FOR REST OF
011D 7E         VP6:    MOV     A,M             ;SCREEN BLANK
011E FE20               CPI     ' '
0120 C22E01             JNZ     VP7
0123 23                 INX     H
0124 7C                 MOV     A,H
0125 FED0               CPI     0D0H    ;END OF SCREEN?
0127 C21D01             JNZ     VP6     ;NOT YET.
012A E1                 POP     H       ;RESTORE HL REGISTERS
012B C33D01             JMP     VP8     ;RETURN TO CP/M. NO MORE TO PRINT
012E E1         VP7:    POP     H               ;RESTORE HL REGISTERS
012F 4E         VP4:    MOV     C,M
0130 CD4C01             CALL    PRINTER
0133 23                 INX     H
0134 7C                 MOV     A,H
0135 FED0               CPI     0D0H
0137 C21301             JNZ     VP3
013A CD4101             CALL    CRLF
013D E1         VP8:    POP     H       ;RESTORE REGS
013E C1                 POP     B
013F C7                 RST     0       ;RETURN TO CP/M
0140 C9                 RET
0141 0E0D       CRLF    MVI     C,0DH           ;CR
0143 CD4C01             CALL    PRINTER
0146 0E0A               MVI     C,0AH           ;LF
0148 CD4C01             CALL    PRINTER
014B C9                 RET
014C 3A0AFE     PRINTER:        LDA     0FE00H+0AH
014F E601                       ANI     01
0151 CA4C01                     JZ      PRINTER
0154 79                         MOV     A,C
0155 3202FE                     STA     0FE00H+2
0158 C9                         RET
0159                    END
```

*Listing 1. Video HARDCOPY routine.*

### Adding Hard Copy

We don't want the request for a hard copy to change the screen. Since we want to copy the screen, a command that echoes to the screen will be messy. We also want to have the option of making a hard copy even when running another program. As long as it looks at the keyboard once in a while, this can be done. This is feasible if all I/O goes through CP/M or your monitor. In this case, we are only concerned about I/O with the keyboard.

With this situation we can have the "keyboard read" routine check for a control key. If it is not the proper key, then process normally. If it is the key that

we have chosen for a hard-copy request, then the logic will branch to the HARDCOPY routine before returning to the calling program.

If you have CP/M and understand the workings of your CBIOS (the Basic I/O Section, Converted for your system), you may be saying, "But I don't have enough room at the top to add a routine to my CBIOS!"

Well, yes you do! And here is how to get more room: Move the entire CP/M system down one kilobyte with your MOVCPM command.

For instance, say you have 24K. Use EDIT to change the EQU for system size in both your CBIOS and your boot loader to 23K. After you add the HARDCOPY routine to your CBIOS, assemble both the boot loader and the CBIOS. Then follow your normal procedures with your MOVCPM command and SYSGEN to create a new version for 23K with hard-copy capabilities.

If you only have a 16K system, you can't SYSGEN a 15K system because 16K is the minimum for CP/M. But try assembling your CBIOS at 16K with the HARDCOPY routine in it. Maybe you will not go beyond your available RAM (i.e., address 3FFF). If you are close, then maybe you can cut some bytes by changing some logic of other parts of your CBIOS.

### Source Listing

The source listing of Listing 3

*Listing 3. CBIOS source listing.*

```
0017 =          MSIZE           EQU     23          ;SIZE OF OPERATING SYSTEM IN KILOBYTES
F800 =          CONTROLLER      EQU     0F800H
FC00 =          BUFF            EQU     CONTROLLER+400H
5A00 =          LOCATION        EQU     MSIZE*1024-512  ;ORG LOCATION FOR THE CBIOS
                ;
5A00                    ORG     LOCATION            ;BASE OF BIOS IN 23K SYSTEM
                ;
0003 =          IOBYTE          EQU     0003H       ;IOBYTE FOR I/O CONTROL
                ;
                ;       < SECTION NOT SHOWN >
                ;
                ;
1C00 =          CBASE   EQU     (MSIZE-16)*1024 ;BIAS FOR SYSTEMS GREATER THAN 16K
4500 =          CPMB    EQU     CBASE+2900H
4D06 =          BDOS    EQU     CBASE+3106H
4480 =          CCPM    EQU     CPMB-128
1500 =          CPML    EQU     $-CPMB
                ;
5A00 C3125B             JMP     COLDBOOT
5A03 C3B95A     EBOOT:  JMP     WBOOT
5A06 C3555B             JMP     CONSTAT
5A09 C3625B             JMP     CONIN
5A0C C3725C             JMP     CONOUT
5A0F C37B5B             JMP     LIST
5A12 C3835B             JMP     PRINTER
5A15 C3955B             JMP     READER
5A18 C32D5A             JMP     HOME1               ;HOME
5A1B C3335A             JMP     TEMPSELDSK          ;SELDSK
5A1E C3515A             JMP     SETTRK1             ;SETTRK
5A21 C30CF8             JMP     SETSEC              ;SETSEC
5A24 C30FF8             JMP     SETDMA              ;SETDMA
5A27 C3985B             JMP     READ                ;DISKREAD
5A2A C3B75B             JMP     WRITE               ;DISKWRITE
                ;
                ;       < SECTION NOT SHOWN >
                ;
                COLDBOOT:               ;DO THIS ONLY ON COLD START
5B12 3E09               MVI     A,00001001B     ;ASSIGN INITIAL I/O
5B14 320300             STA     IOBYTE
5B17 AF                 XRA     A
5B18 D3C8               OUT     0C8H    ;INIT. SCREEN
5B1A CDAA5C             CALL    CLRCRT  ;CLEAR THE SCREEN
5B1D 21115D             LXI     H,SIGNON
5B20 CD185C             CALL    PRMSG   ;PRINT SIGNON MESSAGE
                ;
                BOOT:

                ;       < SECTION NOT SHOWN >

5B52 C30045             JMP     CPMB
                ;
                ;
                ;       CONIN ROUTINE CHECKS FOR FUNCTION KEYS
                ;       IF NOT A FUNCTION, THEN IT RETURNS THE
                ;       KEY IN REGISTER 'A'.
                ;
5B62 CD555B     CONIN:          CALL    CONSTAT
5B65 CA625B                     JZ      CONIN
5B68 3A02FE                     LDA     WRCONT+2
5B6B E67F                       ANI     7FH
```

Program continues.

*Listing 2. HARDCOPY routine in Z-80 code.*

```
                        ;       ADDRESSED AT HEX 100 FOR STAND ALONE
                        ;       TEST UNDER CP/M.
                        ;
CC00                    SCRN    =       0CC00H  ;VIDEO SCREEN ADDRESS
                                .PABS   ;CREATE INTEL HEX FILE
0100                            .LOC 100H
0100    C5                      PUSH B  ;SAVE REGS
0101    E5                      PUSH H
0102    21 CC00                 LXI     H,SCRN  ;SCREEN ADDRESS
0105    7E              VP1:    MOV     A,M     ;LOOK FOR FIRST
0106    FE20                    CPI     ' '     ;NON-BLANK LINE.
0108    2003                    JRNZ    VP2
010A    23                      INX     H
010B    18F8                    JMPR    VP1
010D    7D              VP2:    MOV     A,L
010E    E6C0                    ANI     11000000B  ;SET TO LINE BEGIN
0110    6F                      MOV     L,A
0111    7D              VP3:    MOV     A,L     ;PRINT REST OF SCREEN
0112    E63F                    ANI     00111111B  ;CHECK END OF LINE
0114    2013                    JRNZ    VP4
0116    CD 013A                 CALL    CRLF
0119    E5              VP5:    PUSH    H       ;CHECK FOR REST OF
011A    7E              VP6:    MOV     A,M     ;SCREEN BLANK
011B    FE20                    CPI     ' '
011D    2009                    JRNZ    VP7
011F    23                      INX     H
0120    7C                      MOV     A,H
0121    FED0                    CPI     0D0H    ;END OF SCREEN?
0123    20F5                    JRNZ    VP6     ;NOT YET.
0125    E1                      POP     H       ;RESTORE HL REGISTERS
0126    180E                    JMPR    VP8     ;RETURN TO CP/M.
                                                ;NO MORE TO PRINT
0128    E1              VP7:    POP     H       ;RESTORE HL REGISTERS
0129    4E              VP4:    MOV     C,M
012A    CD 0145                 CALL    PRINT
012D    23                      INX     H
012E    7C                      MOV     A,H
012F    FED0                    CPI     0D0H
0131    20DE                    JRNZ    VP3
0133    CD 013A                 CALL    CRLF
0136    E1              VP8:    POP     H       ;RESTORE REGS
0137    C1                      POP     B
0138    C7                      RST     0       ;RETURN TO CP/M
0139    C9                      RET
013A    0E0D            CRLF:   MVI     C,0DH           ;CR
013C    CD 0145                 CALL    PRINT
013F    0E0A                    MVI     C,0AH           ;LF
0141    CD 0145                 CALL    PRINT
0145    3A FE0A         PRINT:  LDA     0FE00H+0AH
0148    E601                            ANI     01
014A    28F9                            JRZ     PRINT
014C    79                              MOV     A,C
014D    32 FE02                         STA     0FE00H+2
0150    C9                              RET
                                .END

CRLF    013A        PRINT   0145        SCRN    CC00
VP1     0105        VP2     010D        VP3     0111
VP4     0129        VP5     0119        VP6     011A
VP7     0128        VP8     0136        .BLNK. 0000:03 X
.DATA. 0000*   X    .PROG. 0000'   X
```

Listing 3 continued.

```
5B6D FE60                      CPI     '`'      ;TAB KEY
5B6F C2755B                    JNZ     MAYBHARD
5B72 3E09                      MVI     A,09     ;TAB CODE
5B74 C9                        RET
5B75 FE7E        MAYBHARD:     CPI     7EH      ;HARD COPY KEY
5B77 C0                        RNZ
5B78 C3255C                    JMP     HARDCOPY         ;GO DO HARDCOPY
                                                        ;THEN GET THE CONSOLE KEY.
5B7B 3A0300      LIST:         LDA     IOBYTE
5B7E E640                      ANI     01000000B        ;CHECK ASSIGNMENT
5B80 C27A5C                    JNZ     CRTOUT
5B83 3A0AFE      PRINTER:      LDA     WRCONT+0AH
5B86 E601                      ANI     01
5B88 CA835B                    JZ      PRINTER
5B8B 79                        MOV     A,C
5B8C FE5C                      CPI     '\'      ;HOLD PRINTER?
5B8E CA625B                    JZ      CONIN    ;CONIN WILL RELEASE ON ANY KEY
5B91 3202FE                    STA     WRCONT+2
5B94 C9                        RET
5B95 3E1A        READER:       MVI     A,1AH    ;FORCE EOF FOR DUMMY READER
5B97 C9                        RET
                 ;
                 ;

                 ;        < SECTION NOT SHOWN >

                 ;
                 PRMSG:  ;PRINT MESSAGE AT H,L TILL '0'
5C18 7E                  MOV     A,M
5C19 B7                  ORA     A        ;ZERO?
5C1A C8                  RZ
5C1B E5                  PUSH    H        ;MORE TO PRINT
5C1C 4F                  MOV     C,A
5C1D CD725C              CALL    CONOUT
5C20 E1                  POP     H
5C21 23                  INX     H
5C22 C3185C              JMP     PRMSG
                 ;
                 ;       VIDEO HARD COPY ROUTINE
                 ;       WRITTEN BY GLENN STOK
                 ;
CC00 =           SCRN    EQU     0CC00H   ;VIDEO SCREEN ADDRESS
5C25 C5          HARDCOPY: PUSH B         ;SAVE REGS
5C26 E5                  PUSH    H
5C27 2100CC              LXI     H,SCRN           ;SCREEN ADDRESS
5C2A 7E          VP1:    MOV     A,M      ;LOOK FOR FIRST
5C2B FE20                CPI     ' '      ;NON-BLANK LINE.
5C2D C2345C              JNZ     VP2
5C30 23                  INX     H
5C31 C32A5C              JMP     VP1
5C34 7D          VP2:    MOV     A,L
5C35 E6C0                ANI     11000000B        ;SET TO LINE BEGINNING
5C37 6F                  MOV     L,A
5C38 7D          VP3:    MOV     A,L              ;PRINT REST OF SCREEN
5C39 E63F                ANI     00111111B        ;CHECK END OF LINE
5C3B C2545C              JNZ     VP4
5C3E CD675C              CALL    CRLF
5C41 E5          VP5:    PUSH    H                ;CHECK FOR REST OF
5C42 7E          VP6:    MOV     A,M              ;SCREEN BLANK
5C43 FE20                CPI     ' '
5C45 C2535C              JNZ     VP7
5C48 23                  INX     H
5C49 7C                  MOV     A,H
5C4A FED0                CPI     0D0H     ;END OF SCREEN?
5C4C C2425C              JNZ     VP6      ;NOT YET.
5C4F E1                  POP     H        ;RESTORE HL REGISTERS
5C50 C3625C              JMP     VP8      ;RETURN. NO MORE TO PRINT
5C53 E1          VP7:    POP     H        ;RESTORE NEXT LINE TO PRINT
5C54 4E          VP4:    MOV     C,M
5C55 CD835B              CALL    PRINTER
5C58 23                  INX     H
5C59 7C                  MOV     A,H
5C5A FED0                CPI     0D0H
5C5C C2385C              JNZ     VP3
5C5F CD675C              CALL    CRLF
5C62 E1          VP8:    POP     H        ;RESTORE REGS
5C63 C1                  POP     B
5C64 C3625B              JMP     CONIN    ;NOW GO GET CHARACTER.
5C67 0E0D        CRLF    MVI     C,0DH            ;CR
5C69 CD835B              CALL    PRINTER
5C6C 0E0A                MVI     C,0AH            ;LF
5C6E CD835B              CALL    PRINTER
5C71 C9                  RET
                 ;
                 ;  VIDEO CONTROL FOR CRT - ROUTINE FOR CP/M BIOS MODULE
                 ;  WRITTEN BY GLENN STOK, STOK COMPUTER INTERFACE
                 ;
004E =           CURS    EQU     4EH      ;REL. CURSOR LOC. STORAGE
00D0 =           SEND    EQU     0D0H     ;END OF SCREEN PAGE.
0040 =           LINE    EQU     64       ;LINE LENGTH
0020 =           BLANK   EQU     20H      ;BLANK
                 ;
5C72 3A0300      CONOUT: LDA     IOBYTE
5C75 E601                ANI     00000001B        ;CHECK I/O ASSIGNMENT
5C77 CA835B              JZ      PRINTER
```

Program continues.

is my CBIOS. Study the CONIN routine, which reads the keyboard and checks for function keys. Note that I wrote the routine to look for a tilde (~) key. This is hex 7E. You can choose another, but make sure you will never need it for anything else. This key becomes a "function key." Maybe your keyboard has function keys that you can use.

### Implementation

Now look at the HARDCOPY routine in my CBIOS. Note the EQU for the screen address of my memory-mapped VDM. Replace this with the proper address for your system.

Most memory-mapped video has 16 lines and 64 characters per line. This is because this size conveniently uses a 1K block of memory (16 times 64 is 1024, or 1K). If your video is not 16 by 64, then this will considerably change the logic I have used in the routine. But maybe you would like to play with it.

Anyway, if you have a 16 by 64 display, let's continue. Insert the code of Listing 1 or 2 into your CBIOS and put the FUNCTION KEY check in your present "keyboard read" routine. Jump to the HARDCOPY routine if the key matches. Correct the call to the PRINTER routine in Listing 1 or 2. You should call your PRINTER routine in your CBIOS.

The HARDCOPY routine saves and restores all affected registers so as not to interfere with any running programs. It will start its transfer to the print device at the beginning of the first non-blank line. It will print up to and including the last non-blank line.

To do this I have it check if the rest of the screen is blank each time it starts a new line. Even though this is redundant, it does not slow down the printer because the printer, being mechanical, is even slower. The alternative is to search, once and for all, before the start and save the ending address to compare with. I decided against this so as not to create an extra burden for anyone who wants to put the routine in a PROM chip.

The ASCII codes recognized by video boards sometimes differ from the codes recognized

MODEL II

26-4002
**64K 1 Drive**
**$3499.00**

# AUTHORIZED TRS-80® DEALER A301

## COMPUTER SPECIALISTS

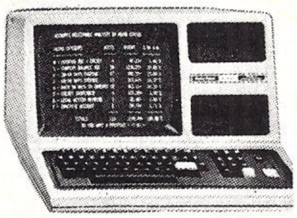
MODEL I

26-1054
**4K Level II**
**$552.00**

| | |
|---|---|
| 26-1056 16K Level II System with Keypad. | $670.00 |
| 26-1145 RS-232 Board | 84.00 |
| 26-1140 "O" K Interface | 249.00 |
| 26-1141 "16" K Interface | 365.00 |
| 26-1142 "32" K Interface | 476.00 |
| 26-1160 Mini Disk - Drive O | 424.00 |
| 26-1161 Mini Disk - Additional | 424.00 |
| 26-1154 Lineprinter II | 720.00 |
| 26-1156 Lineprinter III | 1799.00 |
| 26-1180 Voice Synthesiser | 339.00 |
| 26-1181 VOXBOX | 145.00 |
| 26-1104 Factory Upper/Lower Case Modifcation Installed | 70.00 |
| 26-1506 Scripsit - Tape | 60.00 |
| 26-1563 Scripsit - Disk | 85.00 |

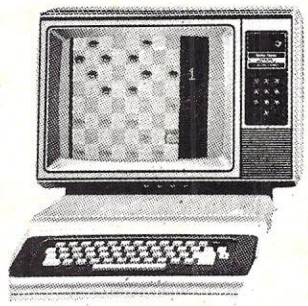
NOTE: Call for availability of VIDEO TEX, Model III, Color, and other new products.

MODEL III



| | |
|---|---|
| 26-1061 4K I | $630.00 |
| 26-1062 16K III | 900.00 |
| 26-1063 32K III 2-Drives, RS232 | 2246.00 |

COLOR



| | |
|---|---|
| 26-3001 4K | $360.00 |
| 26-3002 16K | 540.00 |
| 26-3010 Color Video | 360.00 |
| 26-1206 Recorder | 54.00 |
| 26-3008 Joysticks | 22.50 |

ALL OTHER R.S. SOFTWARE FURNITURE, STANDS, CABLES AND ACCESSORIES DEDUCT 10% FROM CATALOG PRICE

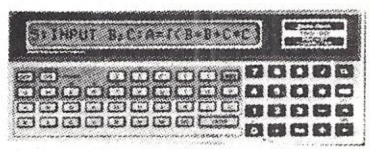
## CENTRONICS

| | |
|---|---|
| Fast 100 CPS Centronics 730 Printer | $675.00 |
| Text Quality Centronics 737 Printer | $850.00 |

**Model II Cobol Compiler $360.00**
**Cobol Run Time Package $36.00**

| | |
|---|---|
| Novation Cat Modem | $149.00 |
| CCA Data Management System | 72.00 |
| Adventure Games Games 1-9 each | 14.00 |

Pocket Computer



| | |
|---|---|
| 26-3501 1.9K P.C. | $225.00 |
| 26-3503 Cassette I/F | 45.00 |
| 14-812 Recorder | 72.00 |

## Acorn Software Products, Inc.

| | |
|---|---|
| GAMES: | |
| Alien Invasion | $9.00 |
| Stock Market | 9.00 |
| Star Trek | 9.00 |
| Block 'Em | 9.00 |
| Ting-Tong | 9.00 |
| UTILITIES: | |
| System Savers | 14.00 |
| EDUCATION: | |
| Language Teacher | 18.00 |

**FREE: PRICE LIST UPON REQUEST**

# 1-800-841-0860 Toll Free Order Entry

✓100

# MICRO MANAGEMENT SYSTEMS, INC.

No Taxes on Out Of State Shipments

Immediate Shipment From Stock on Most Items

DOWNTOWN PLAZA SHOPPING CENTER
115 C SECOND AVE. S.W.
CAIRO, GEORGIA 31728
**(912) 377-7120 Ga. Phone No.**

Full Factory Warranty on All Items Sold.

Largest Inventory In the S.E. U.S.A.

*TRS-80 is a registered trademark of the Tandy Corp.

Listing 3 continued.

```
5C7A 2A4E00    CRTOUT: LHLD    CURS
5C7D EB                 XCHG            ;PUT RELATIVE CURSOR IN D
5C7E 2100CC             LXI    H,SCRN  ;PUT SCREEN ADDRESS IN HL
5C81 19                DAD     D       ;GET ABSOLUTE CURSOR LOC.
5C82 41                MOV     B,C
5C83 7E                MOV     A,M
5C84 E67F              ANI     7FH     ;TURN OFF CURSOR
5C86 77                MOV     M,A
5C87 78                MOV     A,B     ;CHECK FOR CONTROL CHARS.
5C88 FE0C              CPI     0CH     ;ERASE SCREEN
5C8A CAAA5C            JZ      CLRCRT
5C8D FE07              CPI     07
5C8F CAA45C            JZ      RINGBELL
5C92 FE08              CPI     08      ;CTL H (BACK SPACE)
5C94 CABC5C            JZ      BS
5C97 FE0A              CPI     0AH     ;LINE FEED
5C99 CAC15C            JZ      LF
5C9C FE0D              CPI     0DH     ;CARRIAGE RETURN
5C9E CAF35C            JZ      CR
5CA1 C3FB5C            JMP     OTHER   ;PUT OUT ANY OTHER CHAR. AS IS.
               ;
5CA4 0E07      RINGBELL: MVI   C,07
5CA6 CD835B            CALL    PRINTER
5CA9 C9                RET
5CAA 2100CC    CLRCRT  LXI     H,SCRN  ;CLEAR THE SCREEN
5CAD 3620      CLR     MVI     M,BLANK ;BLANK OUT THIS LOCATION.
5CAF 23                INX     H
5CB0 7C                MOV     A,H
5CB1 FED0              CPI     SEND    ;END OF SCREEN?
5CB3 C2AD5C            JNZ     CLR
5CB6 210000            LXI     H,0     ;HOME THE CURSOR.
5CB9 C3015D            JMP     PUT
5CBC 1B        BS      DCX     D
5CBD EB                XCHG
5CBE C3015D            JMP     PUT
               ;
5CC1 214000    LF      LXI     H,LINE  ;LINE FEED
5CC4 19                DAD     D       ;ADD LINE LENGTH TO REL. CURSOR
5CC5 CDCB5C            CALL    SCROLL  ;DO SCROLL IF END OF SCREEN
5CC8 C3015D            JMP     PUT
5CCB 7C        SCROLL  MOV     A,H
5CCC FE04              CPI     4       ;CHECK IF PAST THE SCREEN AREA.
5CCE D8                RC              ;DO NOTHING IF NOT
5CCF E5                PUSH    H
```

by printers. Most printers respond to true ASCII codes, but some video boards have graphics abilities. To tell printable characters from graphics, these boards may use the high-order bit (usually for parity) to trigger the graphics representation of a byte. To print a printable character, this bit may have to turn on or off. You could add an OI or ANI to OR or AND, respectively, each byte before calling your PRINTER routine.

Check if this is necessary in your case. VDM does not require this, but the Polymorphic video display may have to have the parity bit shut off (ANI 7FH) for some printers to recognize the byte as the proper character. As for graphics, you'll have to sub-

```
5CD0 1100CC             LXI    D,SCRN  ;START SCROLL FROM THE TOP
5CD3 2140CC             LXI    H,SCRN+LINE     ;SET HL TO SECOND LINE
5CD6 7E        SWAP     MOV    A,M     ;GET CHAR.
5CD7 23                 INX    H
5CD8 EB                 XCHG           ;GET ADDRESS OF LINE ABOVE.
5CD9 77                 MOV    M,A     ;PUT THE CHAR. THERE
5CDA 23                 INX    H
5CDB EB                 XCHG
5CDC 7C                 MOV    A,H
5CDD FED0               CPI    SEND    ;IS SCREEN FINISHED?
5CDF C2D65C             JNZ    SWAP
5CE2 EB                 XCHG
5CE3 0620               MVI    B,BLANK ;BLANK THE LAST LINE
5CE5 70        LAST     MOV    M,B
5CE6 23                 INX    H
5CE7 7D                 MOV    A,L
5CE8 FE00               CPI    0
5CEA C2E55C             JNZ    LAST
5CED E1                 POP    H       ;GET BACK REL. CURSOR
5CEE 11C0FF             LXI    D,0-LINE
5CF1 19                 DAD    D       ;MOVE UP ONE LINE.
5CF2 C9                 RET
               ;
5CF3 3EC0      CR       MVI    A,11000000B     ;GO TO BEGINNING OF LINE.
5CF5 A3                 ANA    E
5CF6 5F                 MOV    E,A
```

stitute nonprintable characters with a printable one. My VDM doesn't have graphics, so I had no need for this routine. Also note that if a blank is not represented by a hex 20 with your video board, then you'll have to correct the CPI ' ' instructions in listings 2 and 3.

I'll be glad to help with any questions, if you help me by including a stamped, addressed envelope.

Enjoy the new power you now have at the touch of a key. But be sure that there's paper in that printer! ■

```
5CF7 EB                 XCHG
5CF8 C3015D             JMP     PUT
                ;
5CFB 70         OTHER   MOV     M,B     ;DISPLAY CHAR.
5CFC 13                 INX     D       ;INCR. CURSOR
5CFD EB                 XCHG
5CFE CDCB5C             CALL    SCROLL  ;SCROLL IF NECESSARY
5D01 7C         PUT     MOV     A,H     ;PUT CURSOR IN NEW LOCATION ON SCREEN
5D02 E603               ANI     3       ;MAKE RELATIVE AGAIN
5D04 67                 MOV     H,A
5D05 224E00             SHLD    CURS    ;UPDATE CURSOR
5D08 1100CC             LXI     D,SCRN
5D0B 19                 DAD     D
5D0C 7E                 MOV     A,M     ;GET CHAR. UNDER CURSOR
5D0D F680               ORI     80H     ;SET CURSOR ON
5D0F 77                 MOV     M,A
5D10 C9                 RET
                ;
5D11 0D0A0A0A   SIGNON: DB      0DH,0AH,0AH,0AH
5D15 2020202020         DB      '         STOK CP/M SYSTEM - VERSION 5/10/79'
5D42 0D0A00             DB      0DH,0AH,0
5D45                    END
```

# Bridging the 1 pF To 100,000 uF Gap

*Take the guesswork out of capacitor values with this inexpensive digital capacitance meter.*

*Robert J. Stetson*
*Applications Engineer*
*BASF Systems*
*Crosby Dr.*
*Bedford, MA 01730*

Defective capacitors can smoke expensive components in your projects and cause all kinds of general grief. Nothing makes our systems act more demon possessed than a problem capacitor deep in the bowels of an otherwise perfect machine. Before you call in the exorcist, build this digital capacitance meter and start verifying capacitors before they go into the system. Later, if they fail, you will have the means to root them out.

The digital capacitance meter is not only capable of verifying the quality of a known value of capacitor, but also can unmask that unmarked capacitor we've all been known to wonder about from time to time.

The limits of this unit span from 1 pF through 100,000 uF. Needless to say, it accommodates all of my needs. The cost is low, too, right around $35. Considering all the unit can accomplish in the way of testing a long range of capacitance values, it's quite easy to use.

In the short time I've had my unit, all the capacitors in my parts cabinet have been tested, and the defective ones have been weeded out. Next, I plan to take advantage of some gigantic unmarked capacitor bonanzas.

### Operating the Digital Capacitance Meter

The control panel layout is shown in Fig. 1. The toggle switch in the lower left-hand corner of the control panel selects between the high- and low-order group of ranges. With the group-select switch in the EH position, the range-control switch selects the group of ranges E-1 through E3. The switch in the EL position causes the range selector to operate from E-6 through E-2.

The ranges identified by the symbols E-6 through E3 are abbreviated expressions in scientific notation. The converted expressions are listed in Table 1. The decimal point is adjusted as directed by the expression, unless otherwise directed in parentheses.

The range E0 is correct as shown in the readout because it is given as the direct value of the capacitor in microfarads. This range covers the value of 1 uF through 99 uF. In the range E-1, the value is correct as shown in the readout because the internal logic places the decimal point for us, so that the readout reads "X.X". This range, then, covers the values 0.1 through 9.9 microfarads.

In the E-2 range, the decimal is also placed, so that the readout shows ".XX" uF. This is the range .01 uF through .99 uF. All of the other ranges require you to place the decimal according to the value of the exponent. The resulting value will always be in microfarads. For instance, 1 pF is equal to .000001 uF.

The power switch is optional and may be omitted if the unit is not going to be left plugged in when not in use. Upon application of power, the unit has a built-in power on reset. The unit should power on with the ready lamp illuminated and 00 in the readout.

The illuminated ready lamp indicates that the unit is ready to test a capacitor. Select the

| | | |
|---|---|---|
| E3 = | x times $10^3$ | |
| E2 = | x times $10^2$ | |
| E1 = | x times 10 | |
| E0 = | x times 0 | (The value given here is correct as shown in the readout.) |
| E-1 = | x times $10^{-1}$ | (The value given here is correct as shown in the readout.) |
| E-2 = | x times $10^{-2}$ | (The value given here is correct as shown in the readout.) |
| E-3 = | x times $10^{-3}$ | |
| E-4 = | x times $10^{-4}$ | |
| E-5 = | x times $10^{-5}$ | |
| E-6 = | x times $10^{-6}$ | |

*Table 1.*

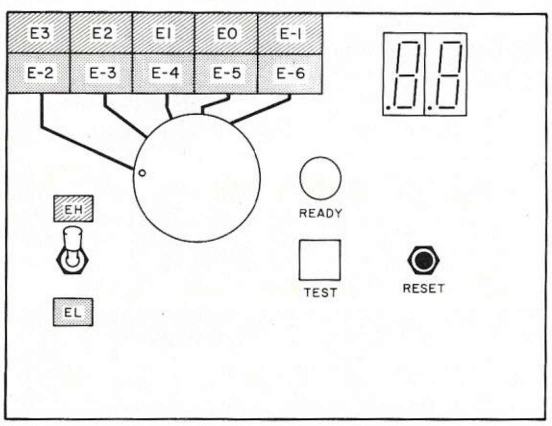



*Fig. 1. Control panel layout.*

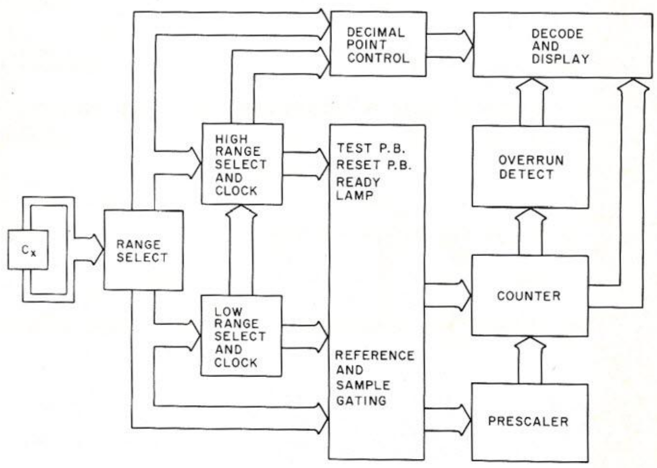



*Fig. 2. Circuit operation.*

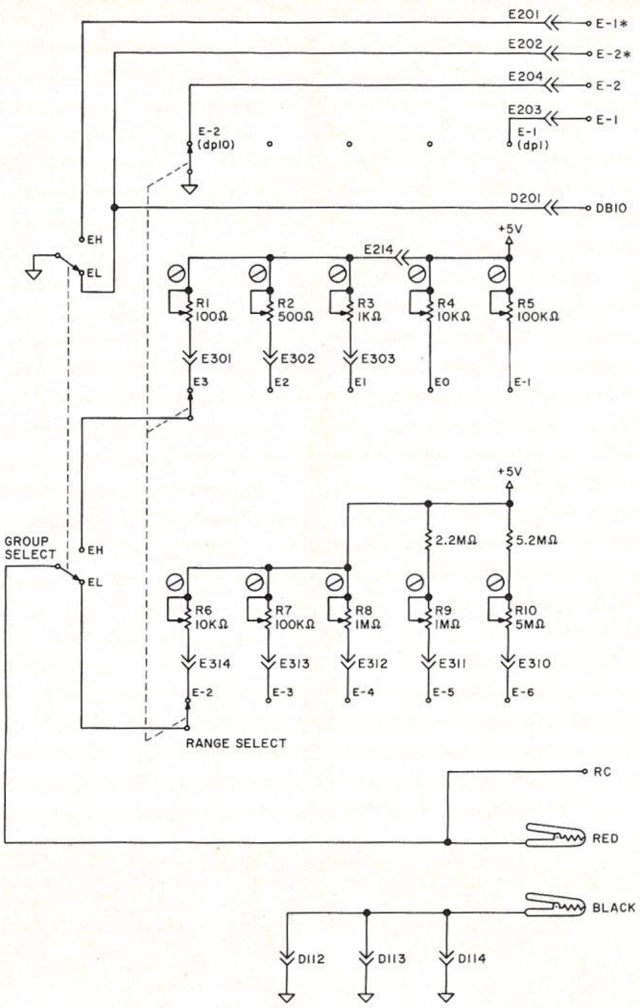



*Fig. 3. Range control.*

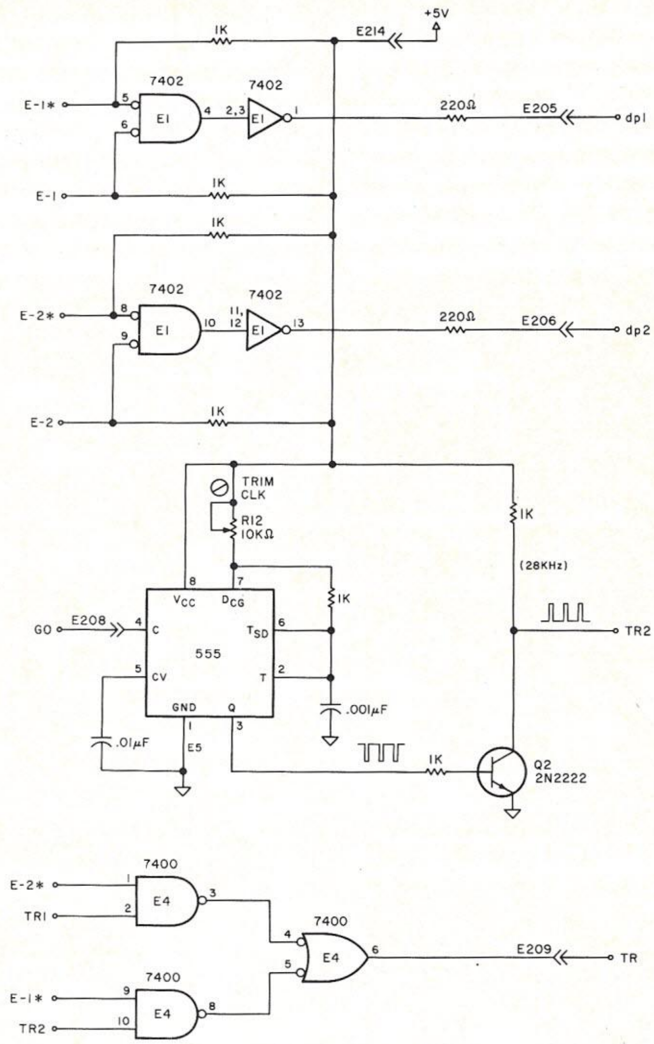



*Fig. 4. Timing and control.*

best range for the capacitor to be tested and connect the black clip to the negative lead (if it is an electrolytic) and the red clip to the positive lead. Press the test push button and the ready lamp goes out. Assuming that the capacitor is good and that the correct range was selected, a number appears in the readout. This, along with the exponential shift, is the value in uF.

If you select the correct range, but the value in the readout is much higher than that shown on the cap, it's leaky. Also, a leaky cap may cause the display to go into an overrange condition, causing the readout to flash on and off. If the cap is shorted, the readout will flash on and off and the counter will continue to count.

### Theory of Operation

An overall view of the circuit operation is shown in Fig. 2. The unit has two groups of ranges, each with its own clock. Each group, in turn, has five ranges.

In Fig. 3, the capacitor to be tested is connected with the positive end on the red clip and the negative end on the black clip. If the value of the capacitor is, say, .01 uF, the group-select switch should be set on EL (Exponential Low range), and the range-select switch should be set to E-2. This setting on the group-select switch will cause the far left decimal to light, and the readout will show .00 when the unit is powered up.

In this situation, the group-select switch (top section) pulls E-2* to ground, and the range-select switch (top section) pulls E-2 to ground. The combination of E-2 and E-2* in Fig. 4 causes dp2 to go low. The signal dp2 goes to Fig. 5, where the decimal point is lit.

The group-select switch also pulls DB10 to ground in the EL position. DB10 goes to Fig. 6, where it presets the 7474 latch. With the latch preset, the $\overline{Q}$ output goes low and enables the 7402 NAND gate (C4) to pass the composite signal generated by combining TR and TX. As long as DB10 remains low, the number of pulses gated through the NAND gate will equal the number of pulses gated out of $\overline{CNT}$. DB10 will remain low as long as the group-select switch is in the EL position.

In Fig. 3, the range-select switch completes a circuit from R6, through the group-select switch to the positive end of the capacitor, to RC, which goes to Fig. 7, where it becomes the timing network to pin 6 and 7 of the 555 timer. The A1 timer (555) is an astable timer, i.e., it fires only once each time it is triggered.

The power-on-reset circuit consists of a 220 uF capacitor from the clear input to the 7474s in Fig. 7 to ground. If the power-on-reset feature is not desired, the 220 uF cap may be eliminated, but the 1k Ohm resistor to +5 volts will still be required as a pull-up. The duration of the reset pulse to the clear input on power up is determined by the RC time constant of the pull-up resistor and the reset capacitor combination. For 220 uF times 1k Ohms, this is .22 seconds.

When the unit powers up, the clear input to the 7474s goes high .22 seconds later than the preset input to ensure the power-up state of the unit. The first latch illuminates the ready lamp on the front panel, while the second latch resets GO and $\overline{SEQ}$.

In Fig. 7, $\overline{SEQ}$ goes high and primes the trigger network to the 555 timer. In the low state, GO holds the other 555 timer re-

set on pin 4 of C1. In Fig. 4, the 555 timer (E5) is also held reset with GO being low on pin 4. In Fig. 6, $\overline{SEQ}$ being high ensures that the divider (or prescaler) is being held at a binary 0000. In Fig. 5, $\overline{SEQ}$ holds the display counter at decimal 00, as well as holding the 7474 overrun detector in a reset condition.

The unit is now ready to provide an accurate test, and the two reference clocks are halted. The low-range reference clock is in Fig. 4 (E5), with the clear held low by GO. The clocks both have inverter-drivers at the outputs to flip the signal from pin 3 of the 555s, so that the signal is normally low with positive-going spikes. Also, the use of transistor driver stages ensures constant loading on the output of the timers and prevents frequency shifts from occurring due to loading or noise coupling at the output.

The high-range reference clock is in Fig. 7 and has the same identical configuration. The only differences in the two clocks are the values placed on the timing components, which generate a low-group selected reference frequency of 28 kHz and a high-group selected reference frequency of 1.5 kHz.

To control the exact number of clock pulses occurring during the timing window, the two clocks are held in a reset state until you press the test button. If the reference clock were free running, the clock pulses would not be uniformly framed in the timing window generated by TX, affecting the unit's accuracy. The unit would be unstable, and the reading would tend to vary from test to test with the same capacitor.

With the capacitor under test in place and the ready lamp lit, press the test push button. The two 7474s in Fig. 7 preset, and the ready lamp extinguishes. GO goes high and $\overline{SEQ}$ goes low, firing the 555 (A1) timer, which generates a single positive-going pulse whose duration is under the control of the capacitor under test and the resistor selected by the group-select switch and the range-select switch combination. $\overline{SEQ}$ also goes to Fig. 6, where the clear releases from the 7490 decade counter, and to Fig. 5, where the 7474 overrun detector has a high applied to the clear and D inputs to enable the detector. Also, $\overline{SEQ}$ going low enables the display counter to display the number of pulses

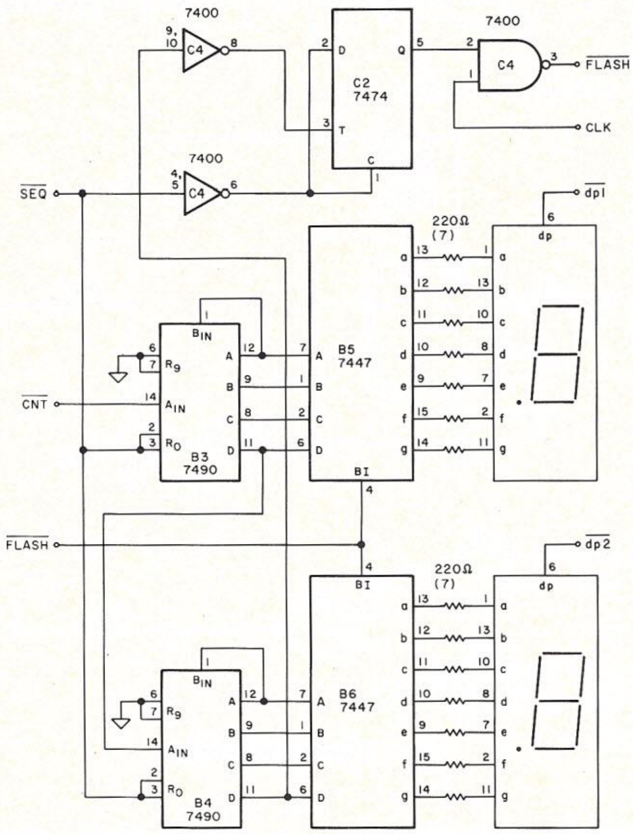



Fig. 5. Counter and output display.

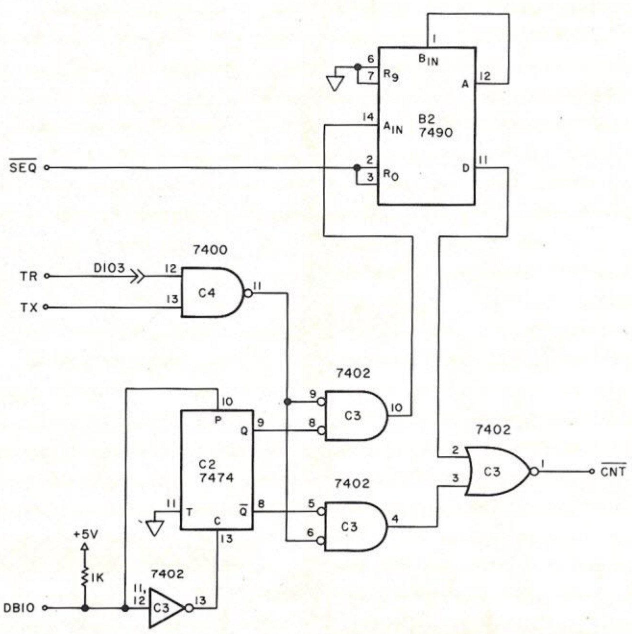



Fig. 6. Prescaler.

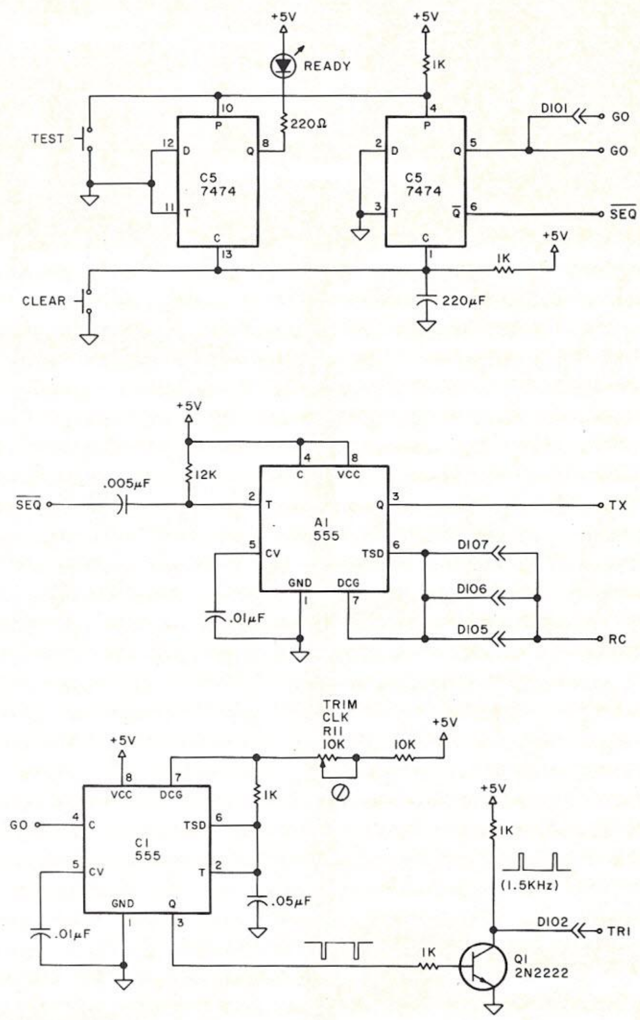



Fig. 7. Timing and control.

that were gated into $\overline{CNT}$.

At the instant that $\overline{SEQ}$ goes low, GO goes high to remove the clear from both the high- and low-range reference clocks. Both the T1 and T2 clocks go to Fig. 4, where the correct clock is selected by E-1* or E-2*. Since the low range is being selected in Fig. 3, E-1* is high and E-2* is low. With E-1* high, the TR2 (low-range clock) is enabled, and TR1 is inhibited by E-1* being low. TR operates at 28 kHz and is gated through Fig. 6 under the control of TX.

If the capacitor is .01 uF, the gating of TR and TX will output one pulse. If a .05 uF capacitor is being tested, the pulse on TX will be five times as long, and five clock pulses will be gated through. Since the prescaler is turned off by DB10, $\overline{CNT}$ is identical to the signal on pin 11 of C4 in Fig. 6.

These pulses are tallied by the display counter in Fig. 5 and directly decoded and displayed on the control panel display. The number showing on the display is the value of the capacitor in microfarads. The decimal, when not lit, is placed by using the exponent indicator selected by the range-select switch.

In the event that the maximum displayed value of 99 is exceeded, the D output of B4 goes low when the 7490 counts up past 9 to 0. This places a positive-going slope on the T input to the 7474 overrange detector.

The output of the 7474 switches high and enables CLK to be gated through to alternately switch $\overline{FLASH}$ high and low. $\overline{FLASH}$ goes to pin 4 of the 7447 binary-to-decimal decoders and alternately enables and disables the outputs *a* through *g*. These outputs are normally high and are prevented from going low while pin 4 is low, so the display flashes on and off. Any time the display flashes, the value being displayed is a random number and *not* the value of the capacitor under test.

The source of the signal CLK is in Fig. 8, where a 74123 dual timer free-runs at all times while power is present, driving CLK on and off for equal 1/2 second intervals. The speed of CLK controls the rate of the flash when an overrange condition is detected.

The power supply in Fig. 8 is simple and direct. The LM 309K is bolted directly to the cabinet from the outside with the leads protruding into the cabinet through two holes. The regulator doesn't even get warm. The transformer is a standard 12 volt filament transformer capable of delivering 500 to 1200 milliamps. A generous amount of bypass capacitors used on the TTL circuits provide complete stability. Every chip is individually bypassed with a .01 uF disk capacitor, and several low-pass 8 uF capacitors were strategically placed on the board.

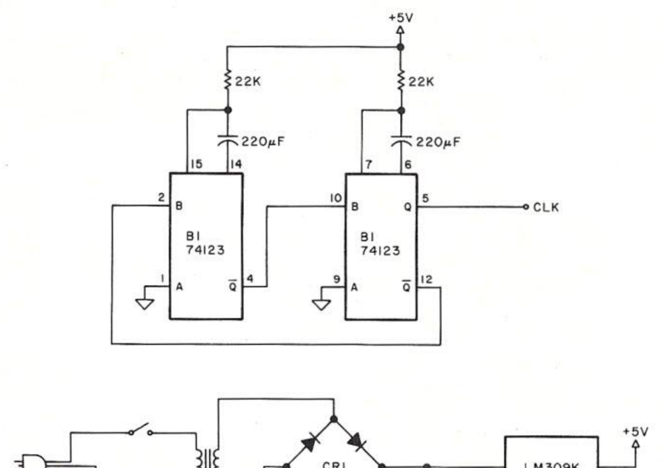



*Fig. 8. Flash clock and power supply.*

Reader Service index—page 241

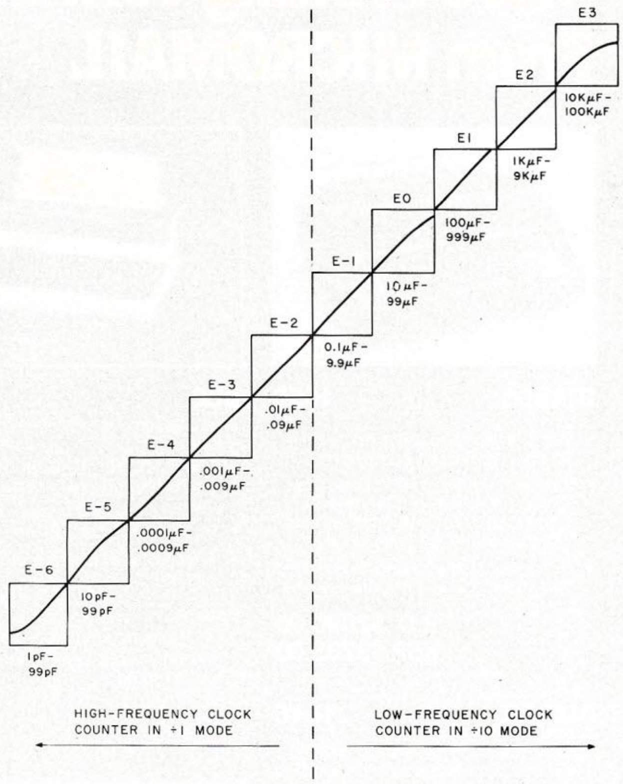



*Fig. 9. Full range linearity map.*

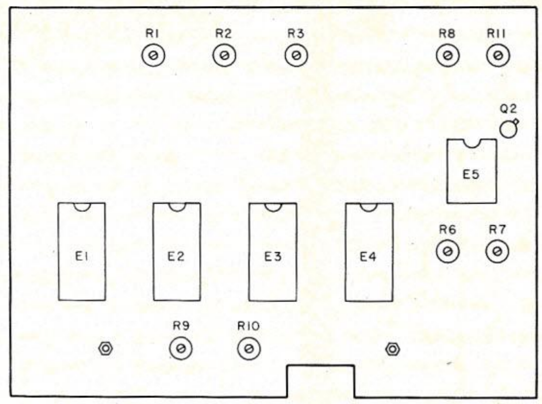



*Fig. 10. PC board layout.*

To test a capacitor of 6.4 uF, the group-select switch is placed in the EH position, and the range-select switch is set to the E-1 position. This places a low on the E-1* line, and DB10 goes high along with E-2*. E-1 and E-1* at the top of Fig. 4 pull dp1 low to Fig. 5 to light the decimal point dp1. The readout now shows 0.0 after the reset button has been pushed.

When the test button is pushed, the sequence of events is similar to the low-range test. There are a couple of differences. First, with E-1* low, TR2 is inhibited and E-2* high enables TR1, so that TR operates at 1.5 kHz. DB10 being high to Fig. 6, the 7474 is held in a cleared condition with Q set low and $\overline{Q}$ set high. The output from C4 pin 11 in Fig. 6 is gated to the 7490 divide-by-10 prescaler. TR and TX gate 640 pulses through to pin 11 of C4. The 7490 decade counter divides the 640 pulses down to 64 pulses on $\overline{CNT}$.

In Fig. 5, the display counter counts up to 64 and stops. With the decimal point dp1 lit, the resultant display is 6.4, which is the value of the capacitor in microfarads.

Since the full range linearity of the 555 timer isn't perfect, linearity problems arise at the upper and lower extremes and the unit becomes nearly impossible to stabilize. For this reason, two clocks were used. A very fast clock (28 kHz) with no prescaler is close to linear all the way down to the tens of pF. Upon nearing the tens of pF, the unit becomes increasingly inaccurate, but still serves as a good go, no-go tester. An accuracy tracking chart is shown in Fig. 9 for all ten ranges in the capacitance spectrum.

Due to the long time delays generated by electrolytic capacitors, a prescaler is used in conjunction with a relatively slow clock. The 1.5 kHz clock aided in stabilizing the erratic upper end of the capacitance spectrum. The accuracy of the unit tracks better than the capacitor under test. The capacitor to be tested usually has a tolerance of 10 to 20 percent. For a 19,000 uF cap, that's a range of 15,000 through 23,000 uF.

Five 19,000 uF caps were sample-tested on the E3 range, and they all tested at 20,000 to 21,000 uF. At values in excess of 50,000 uF, the tolerance of the unit begins to increase, until, at 100,000 uF, the unit's accuracy is 15 percent. A sampling of 100,000 uF capacitors measured 85,000 uF, which is 15 percent below their rated value. The capacitors themselves were rated at 20 percent tolerance, indicating that the unit is well within reasonable limits for a capacitor of this value.

Tracking over a majority of the midrange of the unit is variable, from 1 percent to 5 percent accurate. This was determined by using ten capacitors of several values and tracking the average reading. The tolerance of the capacitors was 20 percent, but the average of a group of ten tracked close to 1 percent. This forms the foundation for my tracking chart and is the method used to determine these percentages.

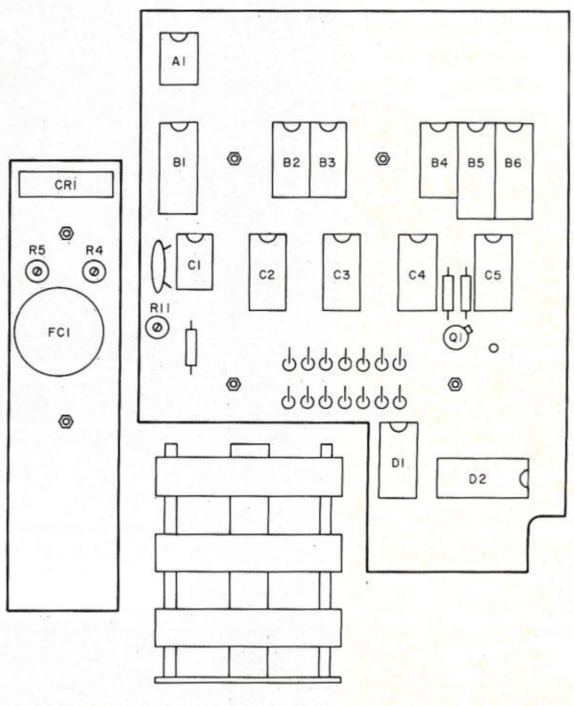



*Fig. 11. PC board layout.*

### Construction

The PC board layout is shown in Figs. 10 and 11. The cabinet was purchased at Radio Shack, along with the common anode seven-segment displays. The wiring was all point-to-point wire wrap, using 30 gauge. No unusual parts were used in the digital capacitance meter. In fact, I selected them specifically for their availability and low cost.

The component values are not critical, and transistor types are unimportant. Simply ensure that they are NPN switching

transistors with sufficient gain to allow reasonable rise and fall time. Try to get as close as possible to the clock frequencies and observe timing-related component values closely. Your particular wiring scheme may cause the calibration to be hard to achieve. If the limit of a calibrated range is reached, add or subtract resistance as required to center the calibration and reestablish stability. This is most likely to vary at the upper and lower extremes of the unit's range (especially toward the E-6 and E3 ranges).

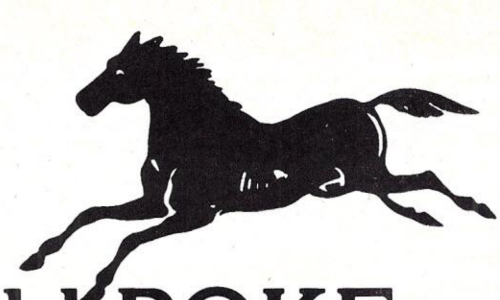
When trimming up each range, favor the end of the range closest to the E0 range for accuracy. Don't allow any single capacitor to set the standard for a range. About ten capacitors give a good indication of how well the range is tracking.

The far left decimal point should light when the E-2 range is selected (make sure the group-select switch is on EL), and the center decimal should be lit when the selector switch is on E-1 (make sure the group-selector switch is on EH). All other ranges should not have any decimal point illuminated. These decimals are set using scientific notation, as derived from the range identifier.

### Testing Capacitors of Unknown Value

The truest value of any unknown capacitor is in the range closest to E0, where a reading is obtained. If the readout shows a value of 00 after the test button has been pushed, the range is too high. Switch to a lower range and try again. If the display flashes on and off with a value indicated, the range is too low. Switch to a higher range and try again. If the display flashes on and off and the display is counting continuously, either you are much too low on the range selected or the capacitor is shorted.

Experience will be your best teacher in using the digital capacitance meter. I have come to trust and rely on mine, which has never failed me yet. ■



# Betting on Old POKEy

*Although you can't always be a winner at the race track, you can cash in on the animated graphics capability of your PET.*

*Gary Greenberg*
*35-63 80th St.*
*Jackson Hts., NY 11372*

One of the nice features of the PET is the POKE command, which places a specific character at a specific location on the PET's screen. This permits you to produce some interesting animated graphics, as illustrated in the Horse Race program in this article.

### Program Operation

POKE puts a given value into a specific location in memory. Lines 1000–1090 show you the format of the POKE instruction. The number after the comma is the value placed into memory. The number before the comma is the address of the memory location in which the number is placed. As you can see from the example, either value can be a variable.

The PET screen accommodates 1000 characters in 25 lines of 40 characters each. The data in certain memory addresses dictate what appears on the screen at any particular location. The 1000 addresses starting consecutively with 32769 determine the screen display. The numeric value stored in address 32769 determines what appears in the first position on the screen (row 1, column 1). The value stored at 32873 determines what appears in the 105th position on the screen (row 3, column 25).

Every PET character has an assigned number that, when stored in one of the 1000 addresses, will display the character at the address indicated. For example, the asterisk is assigned number 42. When 42 is poked into address 32873 (POKE 32873,42), then the screen will display an asterisk at row 3, column 25. The asterisk will remain there until you either poke another value into 32873 or write over that location by running your program.

In the Horse Race program, the POKEs are used to draw the horses and advance them across the screen. The number 32, when poked into memory, produces a blank at the indicated screen location. The POKEs are arranged so that every time a horse is advanced one space, blanks are placed in all the necessary locations to erase all remnants of the old horse drawing.

In line 250 I defined the starting positions for each of the five horses by using the value A(I). As A(I) changes value, the horses are advanced across the screen. Lines 600–610 determine which horse will be moved. Subroutine 1000 causes the horse to be drawn in the proper place on the screen.

This version of Horse Race permits only one player to bet. Each horse has an equal chance of winning. The computer will keep track of the player's winnings. Lines 150–170 are used to provide for a random starting point for the random number generator. The PET uses the same sequence of random numbers every time you turn it on. Since writing this program, I have learned that those lines can be replaced by 150 R = RND( – TI). ■

```
10 PRINT"": GOSUB1200
20 PRINTTAB(8);"COPYRIGHT 1978 BY"
30 PRINTTAB(9);"GARY GREENBERG"
40 GOSUB1200
50 R1=500
80 PRINT"WELCOME TO SILICON VALLEY RACE TRACK.":GOSUB1200
90 PRINT"YOU HAVE $500 TO START WITH."
100 PRINT"ALL HORSES ARE 4:1 ODDSTO WIN.":PRINT
110 PRINT"THE HORSES ARE NUMBERED 1-5."
120 PRINT"ENTER THE NUMBER OF YOUR HORSE AND"
130 PRINT"THEN ENTER YOUR BET."
140 GOSUB1200
150 FORI=1TOVAL(RIGHT$(TI$,2)):R=RND(1):NEXTI
160 T1=INT(1000*RND(1))
170 FORI=1TOT1:R=RND(1):NEXTI
240 FORI=1TOVAL(RIGHT$(TI$,2)):R=RND(1):NEXTI
250 A(1)=2:A(2)=122:A(3)=242:A(4)=362:A(5)=482
260 FORI=1TO5:B(I)=0:NEXTI
270 INPUT"HORSE";H
280 H=INT(H):IFH>0ANDH<6THEN310
290 PRINT"HORSES ARE NUMBERED 1-5. TRY AGAIN"
300 GOSUB1200:GOTO270
310 INPUT"BET";B:B=ABS(B)
320 IFB<R1ORB=R1THEN400
330 PRINT"YOU HAVE $";R1;". ";
350 GOTO310
400 PRINT"";
410 FORI=1TO10:PRINT:IFI/2=INT(I/2)THENPRINTI/2;:GOSUB1200
420 NEXTI
430 FORI=1TO5
440 GOSUB1000
450 NEXTI
600 R=INT(5*RND(1)+1)
610 A(R)=A(R)+1
620 I=R:GOSUB1000
630 B(R)=B(R)+1
640 IFB(R)<34THEN600
650 PRINT"":FORI=1TO15:PRINT:NEXTI
660 PRINT"THE WINNER IS NUMBER";R
670 IFH=RTHENR1=R1+4*B:GOTO750
680 PRINT"YOU LOSE.":R1=R1-B
685 PRINT"YOU HAVE $";R1;"LEFT."
687 IFR1=0THENPRINT"YOU'RE BROKE. COME BACK ANOTHER DAY.":GOTO9999
690 INPUT"ANOTHER RACE (Y OR N)";Z$
700 IF LEFT$(Z$,1)<>"Y"THEN9999
710 GOTO250
750 PRINT"YOU WIN $";4*B
760 PRINT"YOU NOW HAVE $";R1;"."
770 GOTO690
990 GOTO9999
1000 POKE32768+A(I)-1,32
1010 POKE32768+A(I),39
1020 POKE32768+A(I)+1,176+I
1030 POKE32768+A(I)+2,160
1040 POKE32768+A(I)+3,95
1050 POKE32768+A(I)+39,32
1060 POKE32768+A(I)+40,78
1070 POKE32768+A(I)+41,32
1080 POKE32768+A(I)+42,32
1090 POKE32768+A(I)+43,77
1100 RETURN
1200 FORJ=1TO36:PRINT"-";:NEXTJ:PRINT:RETURN
9999 END
READY.
```

*Program listing. Horse Race program in PET BASIC run on a PET printer. The symbols between the quotes in lines 10 and 650 are the clear and home symbols, respectively.*

# The 16-Bit Time Trials

*These assembly-language benchmarks test a new, improved generation of microprocessors.*

*Allan Flippin*
*EECO, Inc.*
*1601 E. Chestnut Ave.*
*Santa Ana, CA 92701*

Since my March 1980 article "Assembly-Language Benchmarks," several new microprocessors have become available. This update will examine four of them: the 8086, 6809, 68000 and Z8000.

I've also included three microprocessors from the first piece: the 370-145 for comparison purposes and the LSI-11/23 and 9900 because both have improved execution speed. I did not recode any of the benchmarks for these microprocessors, but simply plugged in the new execution times from the manufacturers' information. Actual listings for the 8086 and 6809 benchmark programs are included with the article. (Contact the author directly for copies of other listings.)

The benchmarks and scoring methods used are the same as for my original article with the following exceptions:

1. Since my data sample is smaller and more uniform, I have used averages instead of medians for my index calculations.

2. Execution times for the multiplication benchmark are prorated to 16 bits instead of eight bits.

### The Benchmarks

The microprocessor execution times are based on the clock frequencies listed in

```
                      ;
                      ;        8086 TABLE LOOKUP ROUTINE
                      ;
                      ;        37 MICROSECONDS
                      ;         8 INSTRUCTIONS
                      ;        17 BYTES
                      ;
                      ;
0800                  START:   ORG     0800H            ;SET PROGRAM ORIGIN
0800: A0 00 09                 MOV     AL,CHAR          ;GET VALUE TO BE SEARCHED FOR
0803: B9 0A 00                 MOV     CX,LENGTH TABLE1   ;GET NUMBER OF TABLE ENTRIES
0806: BF 01 09                 MOV     DI,OFFSET TABLE1   ;GET TABLE BEGINNING ADDRESS
0809: FC                       CLD                      ;SET UP FOR AUTO-INCREMENT IN 'SCAS' INSTRUCTION
                      ;
080A: F2 AE                    REPNZ
                      &        SCAS    TABLE1           ;SEARCH TABLE UNTIL EITHER WE FIND THE CORRECT
                                                        ;VALUE OR TABLE ENTRY COUNTER GOES TO 0
080C: 75 03                    JNZ     ERROR            ;PROCESS ERROR IF NO MATCH
080E: 8A 45 09                 MOV     AL,TABLE2-TABLE1-1[DI]   ;OTHERWISE, GET CORRESPONDING ENTRY
                                                        ;FROM DATA TABLE
                      ;
                      ;
                      ;-------END OF ROUTINE
                      ;
0812                  ERROR    EQU     $
0812: EB FE           END:     JMP     END              ;LOOP
                      ;
                      ;
                      ;-------DATA AREAS
                      ;
0900                           ORG     0900H
0900                  CHAR:    DB      ?                ;CONTAINS VALUE TO BE SEARCHED FOR
                      ;
0901: 00 01 02 03     TABLE1:  DB      0,1,2,3,4,5,6,7,8,9  ;SEARCH TABLE
      04 05 06 07
      08 09
                      ;
090B: 00 01 02 03     TABLE2:  DB      0,1,2,3,4,5,6,7,8,9  ;CORRESPONDING SEARCH TABLE
      04 05 06 07
      08 09
```

*Listing 1. 8086 table lookup routine.*

```
                        ;
                        ;       8086 BLOCK MOVE SUBROUTINE AND CALLING SEQUENCE
                        ;
                        ;       694 MICROSECONDS
                        ;        11 INSTRUCTIONS
                        ;        21 BYTES
                        ;
                        ;
                        ;-------CALLING SEQUENCE
                        ;
0800                    START   ORG     0800H           ;SET PROGRAM ORIGIN
0800: BE 00 09                  MOV     SI,OFFSET FROM  ;LOAD ADDRESS OF SOURCE FIELD
0803: BF 00 0A                  MOV     DI,OFFSET TO    ;AND ADDRESS OF DESTINATION FIELD
0806: E8 02 00                  CALL    MOVE            ;CALL MOVE SUBROUTINE
                        ;
                        ;
                        ;-------END OF ROUTINE
                        ;
0809: EB FE             END:    JR      END             ;LOOP
                        ;
                        ;
                        ;-------BLOCK MOVE SUBROUTINE
                        ;
080B: 8B DE             MOVE:   MOV     BX,SI           ;SAVE STARTING ADDRESS OF SOURCE FIELD
080D: FC                        CLD                     ;SET UP FOR AUTO-INCREMENT
                        ;
080E: AC                MOVE1:  LODS    FROM            ;GET A BYTE FROM SOURCE FIELD
080F: AA                        STOS    TO              ;STORE IN DESTINATION FIELD
0810: 3C 0D                     CMP     AL,0DH          ;CHECK TO SEE IF CARRIAGE RETURN JUST MOVED
0812: 75 FA                     JNZ     MOVE1           ;LOOP BACK IF NOT
                        ;
0814: 2B F3                     SUB     SI,BX           ;CALCULATE NUMBER OF BYTES MOVED
0816: C3                        RET                     ;EXIT SUBROUTINE
                        ;
                        ;
                        ;-------DATA AREAS
                        ;
0900                            ORG     0900H
0900                    FROM    DB      256 DUP (?)     ;SOURCE FIELD
0A00                    TO      DB      256 DUP (?)     ;DESTINATION FIELD
```

*Listing 2. 8086 block move subroutine and calling sequence.*

Table 1.

I have added charts to compare the members of a given manufacturer's microprocessor family with each other. I assigned performance indices of "1.000" to the eldest family member for all benchmarks and all categories. Indices for the other family members are then calculated relative to the eldest family member. These charts should indicate the better performance offered by the newer microprocessors.

The 8086 is the overall winner, doing well in all categories. It excels in memory utilization. This is because its architecture accommodates one- and three-byte instructions. In many cases, one of these can do the same task as a corresponding two- or four-byte instruction on another 16-bit microprocessor. The result is compact code, requiring about 20 percent less memory space than either the 68000 or Z8000.

The 8086 also is fast, although it is slower than the 68000, which runs at the same clock frequency. Special-purpose string processing instructions make the table lookup the 8086's fastest benchmark. The 16-bit multiply is its slowest benchmark. It requires 70 percent more time than the 68000.

In the ease of programming category, the 8086 is on par with the other microprocessors, but these figures don't mean much since the variations between the microprocessors are so small. I feel that the 8086 is more difficult to program than the 68000, LSI-11/23 or 9900. One reason is that its registers, though more versatile than the 8080's, are not completely general-purpose. Many of the instructions pertain to specific registers and cannot be used with other registers. Also, special-purpose instructions such as those used for string processing increase the number of rules that a programmer must remember in order to produce working code.

The 68000, though number two in the overall ratings, is number one in execution speed. Even though the 68000 and 8086 have the same clock frequency and similar memory timing, instructions tend to execute quicker on the 68000. For example, an instruction to load a 16-bit value from memory into a data register takes 1.5 microseconds on the 68000. A similar instruction will take 1.875 microseconds executed on the 8086.

In my opinion the 68000 is one of the easiest to program. Instead of using special-purpose instructions for string manipulation, the 68000 handles these tasks efficiently with Move and Compare instructions, using the Post-Increment and Pre-Decrement addressing modes.

The 68000's memory utilization is only mediocre. This is the price for having a large number of registers and addressing modes. With memory prices dropping rapidly, perhaps memory utilization is no longer an important criterion for microprocessor selection. Curiously, the 6809 performs signifi-

```
                         ;
                         ;      8086 JUMP TABLE ROUTINE
                         ;
                         ;       6 MICROSECONDS
                         ;       3 INSTRUCTIONS
                         ;      10 BYTES
                         ;
                         ;
0800                     START  ORG    0800H              ;SET PROGRAM ORIGIN
0800: 8B 36 00 09               MOV    SI,STATE           ;LOAD STATE WORD
0804: D1 E6                     SHL    SI,1               ;MULTIPLY BY 2
0806: FF A4 02 09               JMP    JMPTBL,[SI]        ;GET JUMP TABLE ENTRY AND JUMP WHERE IT POINTS
                         ;
                         ;
                         ;-------END OF ROUTINE
                         ;
080A: EB FE              END:   JMP    END                ;LOOP
                         ;
                         ;
                         ;-------DATA AREAS
                         ;
0900                            ORG    0900H
0900                     STATE  DW     ?                  ;STATE WORD
                         ;
0902: 0A08 0A08          JMPTBL DW     END,END,END,END,END,END  ;JUMP TABLE
      0A08 0A08
      0A08 0A08
```

*Listing 3. 8086 jump table routine.*

```
                         ;
                         ;      8086 MULTIPLY ROUTINE
                         ;
                         ;      19 MICROSECONDS
                         ;       2 INSTRUCTIONS
                         ;       7 BYTES
                         ;
                         ;
0800                     START  ORG    0800H              ;SET PROGRAM ORIGIN
0800: A1 00 09                  MOV    AX,NUM1            ;GET MULTIPLICAND
0803: F7 26 02 09               MUL    AX,NUM2            ;MULTIPLY BY MULTIPLIER AND LEAVE RESULT IN
                                                          ;REGISTERS AX AND DX
                         ;
                         ;
                         ;-------END OF ROUTINE
                         ;
0807: EB FE              END:   JMP    END                ;LOOP
                         ;
                         ;
                         ;-------DATA AREAS
                         ;
0900                            ORG    0900H
0900                     NUM1   DW     ?                  ;MULTIPLICAND
0902                     NUM2   DW     ?                  ;MULTIPLIER
```

*Listing 4. 8086 multiply routine.*

cantly better than the 68000 in this category.

The Z8000 slips behind both the 8086 and 68000 in execution time. The main reason for this is the Z8000's slow clock speed and memory timing. Memory cycles require 750 ns, compared to 500 ns for the 6809, 68000 and 8086.

The Z8000's best category is ease of programming. It requires fewer instructions to program my benchmarks than the other microprocessors. However, because of the "special case" nature of the Z8000's instruction set, it (like the Z-80) is actually much more difficult to program than the fig-

```
                        *
                        *       6809 TABLE LOOKUP ROUTINE
                        *
                        *       84 MICROSECONDS
                        *        8 INSTRUCTIONS
                        *       21 BYTES
                        *
                        *
4000                    START   ORG     $4000           SET PROGRAM ORIGIN
4000: B6 50 00                  LDA     BYTE            GET BYTE TO BE SEARCHED FOR
4003: B7 50 0B                  STA     TABEND          SAVE IN DUMMY TABLE ENTRY
4006: 8E 50 01                  LDX     #TABLE1         GET SEARCH TABLE ADDRESS
                        *
4009: A1 80             SEARCH  CMPA    ,X+             CHECK FOR MATCH AND INCREMENT TABLE POINTER
400B: 26 FC                     BNE     SEARCH          LOOP BACK IF NOT
                        *
400D: 8C 50 0B                  CMPX    #TABEND         CHECK FOR MATCH ON DUMMY ENTRY
4010: 22 03                     BHI     ERROR           GO PROCESS ERROR IF SO
4012: A6 88 0A                  LDA     OFFSET-1,X      OTHERWISE, GET CORRESPONDING ENTRY FROM DATA TABLE
                        *
                        *
                        *-------END OF ROUTINE
                        *
4015                    ERROR   EQU     *
4015: 20 FE             END     BRA     END             LOOP
                        *
                        *
                        *-------DATA AREAS
                        *
5000                            ORG     $5000
5000: 00                BYTE    FCB     0               CONTAINS VALUE TO SEARCH FOR
                        *
5001: 00 01 02 03       TABLE1  FCB     0,1,2,3,4,5,6,7,8,9  SEARCH TABLE
      04 05 06 07
      08 09
500B: 00                TABEND  FCB     0               DUMMY ENTRY
                        *
500C: 00 01 02 03       TABLE2  FCB     0,1,2,3,4,5,6,7,8,9  CORRESPONDING DATA TABLE
      04 05 06 07
      08 09
000B                    OFFSET  EQU     TABLE2-TABLE1   OFFSET BETWEEN TABLES
```

*Listing 5. 6809 table lookup routine.*

```
                        *
                        *       6809 BLOCK MOVE SUBROUTINE AND CALLING SEQUENCE
                        *
                        *       1109 MICROSECONDS
                        *         11 INSTRUCTIONS
                        *         25 BYTES
                        *
                        *
                        *-------CALLING SEQUENCE
                        *
4000                    START   ORG     $4000           SET PROGRAM ORIGIN
4000: 8E 42 00                  LDX     #FROM           LOAD ADDRESSES OF SOURCE AND
4003: 10 8E 43 00               LDY     #TO             DESTINATION FIELDS
4007: BD 41 00                  JSR     MOVE            CALL BLOCK MOVE SUBROUTINE
                        *
                        *
                        *-------END OF ROUTINE
                        *
400A: 20 FE             END     BRA     END             LOOP
                        *
                        *
                        *-------DATA AREAS
                        *
4200                    FROM    EQU     $4200           SOURCE FIELD
4300                    TO      EQU     $4300           DESTINATION FIELD
                        *
                        *
                        *-------BLOCK MOVE SUBROUTINE
                        *
4100                            ORG     $4100           SET SUBROUTINE ORIGIN
4100: 34 10             MOVE    PSHS    X               SAVE SOURCE FIELD ADDRESS ON STACK
                        *
4102: A6 80             MOVE1   LDA     ,X+             GET A BYTE FROM SOURCE FIELD AND INCREMENT POINTER
4104: A7 A0                     STA     ,Y+             SAVE IN DESTINATION FIELD AND INCREMENT POINTER
4106: 81 0D                     CMPA    #$0D            CHECK FOR CARRIAGE RETURN JUST MOVED
4108: 26 F8                     BNE     MOVE1           LOOP BACK IF NOT
                        *
410A: 1F 10                     TFR     X,D             MOVE ENDING SOURCE FIELD ADDRESS TO D REG.
410C: A3 E1                     SUBD    ,S++            SUBTRACT BEGINNING SOURCE FIELD ADDRES FROM IT TO
                        *                               CALCULATE NUMBER OF BYTES MOVED
                        *                               (ALSO, TAKE SOURCE FIELD ADDRESS OFF STACK)
410E: 39                        RTS                     EXIT SUBROUTINE
```

*Listing 6. 6809 block move subroutine and calling sequence.*

```
                         *
                         *         6809 JUMP TABLE ROUTINE
                         *
                         *         8 MICROSECONDS
                         *         4 INSTRUCTIONS
                         *         9 BYTES
                         *
                         *
4000                     START     ORG      $4000         SET PROGRAM ORIGIN
4000: B6 50 00                     LDA      STATE         GET STATE BYTE
4003: 48                           ASLA                   MULTIPLY BY 2
4004: 8E 50 01                     LDX      #JMPTBL       LOAD JUMP TABLE ADDRESS
4007: 6E 96                        JMP      [A,X]         GET JUMP TABLE ENTRY AND JUMP WHERE IT POINTS
                         *
                         *
                         *-------END OF ROUTINE
                         *
4009: 20 FE              END       BRA      END           LOOP
                         *
                         *
                         *-------DATA AREAS
                         *
5000                               ORG      $5000
5000: 00                 STATE     FCB      0             STATE BYTE
                         *
5001: 4009 4009          JMPTBL    FCW      END,END,END,END,END,END   JUMP TABLE
      4009 4009
      4009 4009
```

*Listing 7. 6809 jump table routine.*

ures indicate.

The 6809 does surprisingly well considering its 16-bit competition. It consistently outperforms all of the previously tested eight-bit microprocessors in all categories. Its execution time ratings are better than both the LSI-11/23 and the 9900. Its memory utilization is second only to the 8086. This is impressive for what amounts to an upgraded 6800.

However, I must disclaim the 6809's good numerical performance in the multiplication benchmark. Since the 6809 does not have a divide instruction, its real performance in arithmetic processing will be much less than indicated by my benchmark. Even so, the 6809 should perform better in arithmetic processing than the older eight-bit microprocessors.

## Conclusions

All of the microprocessors perform quite well. The overall rating index for the lowest-ranked microprocessor is only 50 percent higher than that for the highest-ranked one. With such a small variance in performance, price may therefore become the deciding factor when choosing between the microprocessors in this comparison.

The new 16-bit microprocessors offer significant improvements in performance over their eight-bit predecessors. On the average, the 16-bit microprocessors executed my benchmarks two to three times as fast as the earlier eight-bit microprocessors. Their benchmarks also required about half as many instructions and 25 percent less memory space. However, as demonstrated by the 6809, eight-bit microprocessors can

| Microprocessor | Clock Frequency/Cycle Time |
|---|---|
| 8086 | 8 MHz clock frequency |
| 68000 | 8 MHz clock frequency |
| Z8000 | 4 MHz clock frequency |
| 6809 | 2 MHz clock frequency |
| LSI-11/23 | 300 ns micro-cycle |
| 9900 | 4 MHz clock frequency |

*Table 1. Clock frequencies.*

```
                         *
                         *         6809 MULTIPLY ROUTINE
                         *
                         *         21 MICROSECONDS (FOR 16 BITS)
                         *          3 INSTRUCTIONS
                         *          7 BYTES
                         *
                         *
4000                     START     ORG      $4000         SET PROGRAM ORIGIN
4000: B6 50 00                     LDA      NUM1          GET MULTIPLICAND
4003: F6 50 01                     LDB      NUM2          GET MULTIPLIER
4006: 3D                           MUL                    MULTIPLY THEM
                         *
                         *
                         *-------END OF ROUTINE
                         *
4007: 20 FE              END       BRA      END           LOOP
                         *
                         *
                         *-------DATA AREAS
                         *
5000                               ORG      $5000
5000: 00                 NUM1      FCB      0             MULTIPLICAND
5001: 00                 NUM2      FCB      0             MULTIPLIER
```

*Listing 8. 6809 multiply routine.*

# PMC-80

## Level II 16K at $645


# SOFTWARE COMPATIBLE

- Reads all Level II BASIC tapes
- Reads all SYSTEM tapes
- Full range of peripherals
- Video output for monitor and TV
- Optional FASTLOAD at 8000 band
- Optional Upper/Lower case

The PMC-80 is a "work-alike" computer to the popular TRS-80* Model I, Level II by Tandy, Radio Shack. The PMC-80 has 16K bytes of RAM and the complete Level II 12K BASIC ROM by Microsoft that makes it 100% software compatible with programs from Radio Shack and from the hundreds of other independent suppliers. The built-in cassette player reads standard Radio Shack programs for the TRS-80.*

Sold through computer stores.

The PMC-80 will operate with any of the many peripherals Radio Shack and other independent vendors have invented to plug into the TRS-80.* Most importantly, the Interface Adapter permits Expansion Interfaces with memory expansion to 48K to be added. An Expansion Interface will also permit the addition of Radio Shack compatible 5¼″ disks and disk operating systems, RS 232, printers, etc.

*TRS-80 is a registered trademark of Tandy, Radio Shack.

## Personal Micro Computers, Inc. ✓112

475 Ellis Street, Mountain View, CA 94043 (415) 962-0220

be made to perform quite well, and actually better than some 16-bit microprocessors.

One of the selling features for the 8086, 68000 and Z8000 is the amount of memory they can access. All three can address millions of bytes, which should be plenty for most hobbyists. However, I limited my addressing range to 65,536 bytes. If more memory is addressed, the code will be larger and slower. ■

| PROCESSOR | TABLE LOOKUP TIME | RANK | INDEX | BLOCK MOVE TIME | RANK | INDEX | JUMP TABLE TIME | RANK | INDEX | MULTIPLY TIME | RANK | INDEX |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8086 | 37 | 1 | .462 | 694 | 2 | .662 | 6 | 2 | .677 | 19 | 3 | 1.000 |
| 68000 | 43 | 2 | .538 | 488 | 1 | .466 | 5 | 1 | .565 | 11 | 2 | .579 |
| Z8000 | 48 | 3 | .600 | 1068 | 4 | 1.019 | 8 | 3 | .903 | 20 | 4 | 1.053 |
| 6809 | 84 | 4 | 1.050 | 1109 | 5 | 1.058 | 8 | 3 | .903 | 21 | 5 | 1.105 |
| LSI-11/23 | 94 | 5 | 1.175 | 1196 | 6 | 1.141 | 10 | 6 | 1.129 | 31 | 7 | 1.632 |
| 370-145 | 109 | 6 | 1.362 | 1027 | 3 | .980 | 8 | 3 | .903 | 10 | 1 | .526 |
| 9900 | 145 | 7 | 1.812 | 1756 | 7 | 1.675 | 17 | 7 | 1.919 | 21 | 5 | 1.105 |
| AVERAGE | 80 | | | 1048.3 | | | 8.9 | | | 19 | | |

*Table 2. Execution time (in microseconds).*

| PROCESSOR | TABLE LOOKUP NO. | RANK | INDEX | BLOCK MOVE NO. | RANK | INDEX | JUMP TABLE NO. | RANK | INDEX | MULTIPLY NO. | RANK | INDEX |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8086 | 8 | 2 | 1.057 | 11 | 2 | 1.013 | 3 | 1 | .778 | 2 | 1 | .933 |
| 68000 | 8 | 2 | 1.057 | 11 | 2 | 1.013 | 4 | 3 | 1.037 | 2 | 1 | .933 |
| Z8000 | 5 | 1 | .660 | 8 | 1 | .737 | 4 | 3 | 1.037 | 2 | 1 | .933 |
| 6809 | 8 | 2 | 1.057 | 11 | 2 | 1.013 | 4 | 3 | 1.037 | 3 | 7 | 1.400 |
| LSI-11/23 | 8 | 2 | 1.057 | 11 | 2 | 1.013 | 3 | 1 | .778 | 2 | 1 | .933 |
| 370-145 | 8 | 2 | 1.057 | 13 | 7 | 1.197 | 5 | 7 | 1.296 | 2 | 1 | .933 |
| 9900 | 8 | 2 | 1.057 | 11 | 2 | 1.013 | 4 | 3 | 1.037 | 2 | 1 | .933 |
| AVERAGE | 7.6 | | | 10.9 | | | 3.9 | | | 2.1 | | |

*Table 3. Ease of programming (in number of instructions).*

| PROCESSOR | TABLE LOOKUP NO. | RANK | INDEX | BLOCK MOVE NO. | RANK | INDEX | JUMP TABLE NO. | RANK | INDEX | MULTIPLY NO. | RANK | INDEX |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8086 | 17 | 1 | .708 | 21 | 1 | .728 | 10 | 2 | .886 | 7 | 1 | .907 |
| 68000 | 26 | 4 | 1.083 | 28 | 4 | .970 | 12 | 4 | 1.063 | 8 | 3 | 1.037 |
| Z8000 | 20 | 2 | .833 | 26 | 3 | .901 | 12 | 4 | 1.063 | 8 | 3 | 1.037 |
| 6809 | 21 | 3 | .875 | 25 | 2 | .866 | 9 | 1 | .797 | 7 | 1 | .907 |
| LSI-11/23 | 26 | 4 | 1.083 | 30 | 5 | 1.040 | 10 | 2 | .886 | 8 | 3 | 1.037 |
| 370-145 | 32 | 7 | 1.333 | 42 | 7 | 1.455 | 14 | 7 | 1.241 | 8 | 3 | 1.037 |
| 9900 | 26 | 4 | 1.083 | 30 | 5 | 1.040 | 12 | 4 | 1.063 | 8 | 3 | 1.037 |
| AVERAGE | 24 | | | 28.9 | | | 11.3 | | | 7.7 | | |

*Table 4. Memory utilization (in bytes).*

## Credits

Thanks to the following people who helped with this update: Jim Howell, for the 6809 code; Don Barnes of Motorola, for the 6809 and 68000 info; and Paul Stapinski of AMD, for the Z8000 info.

For a further look at the 8086, 68000 and Z8000 16-bit microprocessors, see *Kilobaud Microcomputing*. "The 16-Bit Super Processors Are Here," August 1980, p. 26.

| | AVERAGES BY BENCHMARK | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | TABLE LOOKUP | | BLOCK MOVE | | JUMP TABLE | | MULTIPLY | |
| PROCESSOR | RANK | INDEX | RANK | INDEX | RANK | INDEX | RANK | INDEX |
| 8086 | 2 | .742 | 1 | .801 | 1 | .780 | 3 | .947 |
| 68000 | 3 | .893 | 2 | .816 | 2 | .888 | 2 | .850 |
| Z8000 | 1 | .698 | 3 | .886 | 5 | 1.001 | 4 | 1.008 |
| 6809 | 4 | .994 | 4 | .979 | 3 | .912 | 6 | 1.137 |
| LSI-11/23 | 5 | 1.105 | 5 | 1.065 | 4 | .931 | 7 | 1.201 |
| 370-145 | 6 | 1.251 | 6 | 1.211 | 6 | 1.147 | 1 | .832 |
| 9900 | 7 | 1.317 | 7 | 1.243 | 7 | 1.340 | 5 | 1.025 |

*Table 5. Averages by benchmark.*

| | AVERAGES BY CATEGORY | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | EXECUTION TIME | | EASE OF PROGRAMMING | | MEMORY UTILIZATION | | OVERALL | |
| PROCESSOR | RANK | INDEX | RANK | INDEX | RANK | INDEX | RANK | INDEX |
| 8086 | 2 | .700 | 2 | .945 | 1 | .807 | 1 | .817 |
| 68000 | 1 | .537 | 4 | 1.010 | 5 | 1.038 | 2 | .862 |
| Z8000 | 3 | .894 | 1 | .842 | 3 | .958 | 3 | .898 |
| 6809 | 5 | 1.029 | 7 | 1.127 | 2 | .861 | 4 | 1.006 |
| LSI-11/23 | 6 | 1.269 | 2 | .945 | 4 | 1.012 | 5 | 1.075 |
| 370-145 | 4 | .943 | 6 | 1.121 | 7 | 1.266 | 6 | 1.110 |
| 9900 | 7 | 1.628 | 4 | 1.010 | 6 | 1.056 | 7 | 1.231 |

*Table 6. Averages by category.*

| | AVERAGES BY BENCHMARK | | | |
|---|---|---|---|---|
| PROCESSOR | TABLE LOOKUP | BLOCK MOVE | JUMP TABLE | MULTIPLY |
| 8085 | .938 | .947 | .917 | .905 |
| 8086 | .573 | .754 | .472 | .134 |

| | AVERAGES BY CATEGORY | | | |
|---|---|---|---|---|
| PROCESSOR | EXECUTION TIME | EASE OF PROGRAMMING | MEMORY UTILIZATION | OVERALL |
| 8085 | .780 | 1.000 | 1.000 | .927 |
| 8086 | .339 | .458 | .654 | .484 |

*Table 7. 8085, 8086 compared to 8080.*

| PROCESSOR | AVERAGES BY BENCHMARK: TABLE LOOKUP | BLOCK MOVE | JUMP TABLE | MULTIPLY |
|---|---|---|---|---|
| 6809 | .785 | .775 | .481 | .241 |
| 68000 | .714 | .665 | .474 | .192 |

| PROCESSOR | AVERAGES BY CATEGORY: EXECUTION TIME | EASE OF PROGRAMMING | MEMORY UTILIZATION | OVERALL |
|---|---|---|---|---|
| 6809 | .591 | .503 | .618 | .571 |
| 68000 | .303 | .486 | .745 | .511 |

*Table 8. 6809, 68000 compared to 6800.*



| PROCESSOR | AVERAGES BY BENCHMARK: TABLE LOOKUP | BLOCK MOVE | JUMP TABLE | MULTIPLY |
|---|---|---|---|---|
| Z8000 | .719 | .933 | .607 | .219 |

| PROCESSOR | AVERAGES BY CATEGORY: EXECUTION TIME | EASE OF PROGRAMMING | MEMORY UTILIZATION | OVERALL |
|---|---|---|---|---|
| Z8000 | .532 | .445 | .881 | .619 |

*Table 9. Z8000 compared to Z-80.*

# Machine-Language Techniques For the 1802

*Puts zip into the COSMAC VIP.*


*Gerald Strope*
*1504 Strothmore Court*
*Raleigh, NC 27609*


The RCA COSMAC VIP is a single-board processor using the 1802 chip. The processor hardware is well explained in two articles entitled "COSMAC Double Play" in the May 1979 issue of *Kilobaud Microcomputing,* so I will not repeat that here. This article will deal with the use of machine language in the VIP.

### Introduction

The CHIP-8 language that was developed for the COSMAC VIP is quite versatile for use in programming many varied applications. It is my opinion, though, that all CHIP-8 programmers will come to a point where they will want to do some particular function that is not possible in CHIP-8, and will need to write a machine-language subroutine.

This article is a collection of ideas and techniques I have compiled while trying to put machine language to work in the VIP. I hope some of these ideas will be useful to you when you try your first machine-language subroutine. I don't intend to write a tutorial on the CHIP-8 or 1802 instruction set in this article. CHIP-8 is covered well in the COSMAC VIP instruction manual, and I suggest you read Tom Pittman's "A Short Course In Programming" (Netronics, 333 Litchfield Rd., New Milford CT 06776) for 1802 machine-language basics. I will try to explain my programs in enough detail so the reader will be able to understand them, assuming the reader has some background in CHIP-8 and 1802 machine language.

After assembling my COSMAC VIP kit and testing it out, I first wanted to try using some Elf-2 programs I had seen in various magazine articles. I ran into some differences almost immediately in Elf-2 and VIP hardware. The first problem I ran into is that the Elf-2 programs use a 6C instruction to input a byte from the keyboard. The 6C instruction is not valid on the VIP; the VIP uses a keyboard scan technique instead (see Fig. 1).

Another difference is the use of the 64 output instruction in the Elf programs. This output instruction will do nothing on the VIP because it does not have an output indicator. The TV display is used for all output.

| ADDRESS | CODE | STEP | COMMENTS |
|---|---|---|---|
| 0000 | F800BB | 0 | Display=page 0 |
| 03 | F806B2 | | R2=06CF, stack pointer |
| 06 | F8CFA2 | | |
| 09 | F881B1 | | R1=8146, refresh routine |
| 0C | F846A1 | | in ROM |
| 0F | 90B3 | | R3=0015, main program |
| 11 | F815A3 | | register |
| 14 | D3 | | R3=program counter |
| 15 | E269 | | X=R2, turn on TV |
| 17 | 3017 | | branch to 0017 (stop loop) |

*Listing 1.*

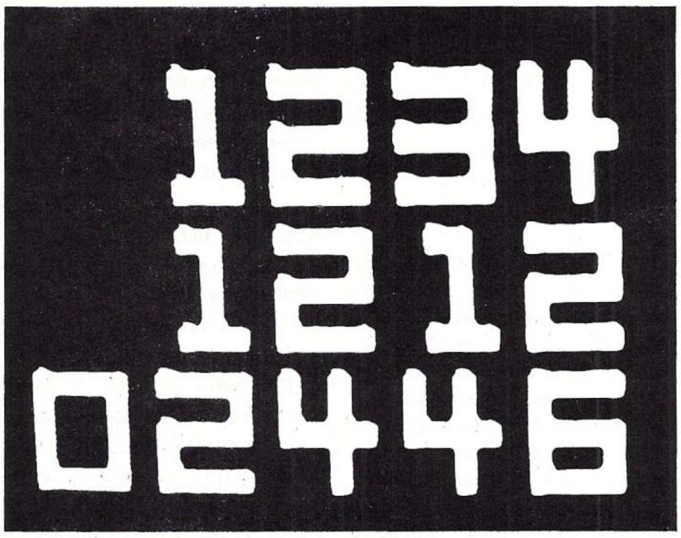
| ADDRESS | CODE | STEP | COMMENTS |
|---|---|---|---|
| 0017 | F800B4 | 1 | put 00 into R4.1, |
| 1A | B5BAAC | | R5.1, RA.1, RC.0 |
| 1D | F827A4 | | R4=0027 |
| 20 | F8C8A5 | | R5=00C8 |
| 23 | F8E0AA | | RA=00E0 |
| 26 | D4E5 | | P=R4, X=R5 |
| 28 | 62 | 2 | m at RX lsd to keyboard latch |
| 29 | 3637 | | branch if EF3=1 |
| 2B | 25F0 | 3 | R5-1, m at RX→D |
| 2D | FC0155 | | add 01, store in m at R5 |
| 30 | F0FB10 | | m at RX→D, OR imm 10-D |
| 33 | 324D | | branch if D=0 |
| 35 | 3052 | | go to 0052 |
| 37 | 25F0 | 4 | R5-1, m at RX→D |
| 39 | 8C3A63 | | RC.0→D, branch if D not 0 |
| 3C | F05A | | m at RX→D, store in m at RA |
| 3E | 3E437B | | wait for key released |
| 41 | 303E | | and turn Q on |
| 43 | 4A2AFE | | m at RA→D, RA-1, shift |
| 46 | FEFEFE | | left four times |
| 49 | 1C5A | | RC+1, store in m at RA |
| 4B | 305A | | go to 005A |
| 4D | FC0055 | 5 | reset m at R5 to 00 |
| 50 | 3028 | | go to 0028 |
| 52 | 0AFBA1 | 6 | m at RA→D, OR imm A1-D |
| 55 | 3A28 | | branch if D not 0 |
| 57 | 7B3028 | | turn on Q, go to 0028 |
| 5A | F820B6 | 7 | load 20 into R6.1 |
| 5D | 26 | | R6-1 |
| 5E | 963228 | | R6.0→D, branch if D=0 |
| 61 | 305D | | go to 005D |
| 63 | 2CF0EA | 8 | RC-1, m at RX→D, X=A |
| 66 | F1 | | M at RA or-D |
| 67 | E55A7B | | X=5, store D in m at RA, Q on |
| 6A | 3E5A | | branch to 005A when key released |
| 6C | 306A | | go to 006A |

*Listing 2.*

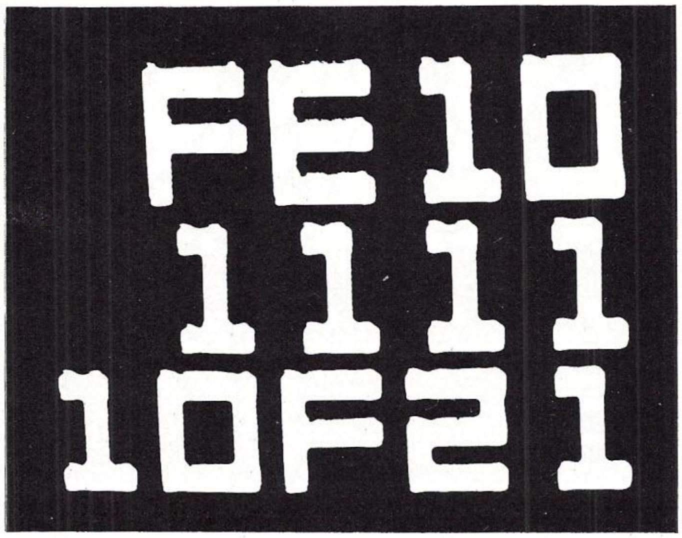
### Inputting and Outputting Data

I will go into methods of solving both of these problems later in this article, but first let's kick off the display so we can see what we are doing. The program in Listing 1 will set up the proper pointers to use the refresh routine in the 512 byte ROM. Register B.1 contains the desired display page (page zero, in this case). Register 2 is the stack pointer. Register 1 points to the refresh routine in the ROM. Register 3 is the main program register. There is a branch to itself in position 0017 that we will remove when we add on to this program in the next step.

Now page 0 of RAM is displayed on the screen. This is convenient because now we will enter a program into page 0, and you can watch a scan loop for the keyboard increment and see a shift instruction operate. If




*The addition program with an answer without a carry from the high order position.*




*The addition program with an answer with a carry from the high order position.*

you forget a step, as I have done, you can watch your program change and disintegrate before your very own eyes. If you see this happening, hit the reset quickly before the refresh pointers are wiped out. If that happens, you have to go back to 0000 and enter it all again.

With the program displayed on the screen, it is sometimes easy to see if a register is incrementing when you don't want it to: Bytes of data will fill up the screen until they crash into part of the program, or some other similar indication.

Listing 2 will scan the keyboard, input a byte of data, place this byte of data in memory location 00E0 and check to see if the byte = hex A1. If it equals A1, turn on the Q light.

The program works as follows:

Step 1—Initialize.

Step 2—Load the least significant digit (LSD) of memory at register 5 into the keyboard latch and test for an EF3 = 1.

Step 3—Scan 00-10 through memory at register 5. When it equals 10, branch to step 5.

Step 4—Program is at this step because an EF3 = 1 occurred. Take memory at R5 location (this is equal to the key pressed) and store in memory at register A. Wait for key released, then shift data into the left half of the byte and store back at RA memory location. Increment register C, which acts as a flip-flop indicator to determine if the left or right half of the byte is being entered. Branch to debounce routine at step 7. This wastes time to make sure the key is fully up before turning the program loose to start scanning again.

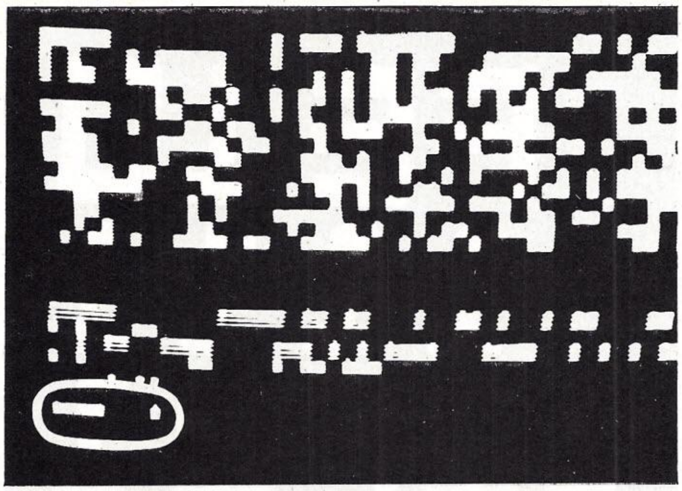
Step 5—Reset memory at R5 location to 00.

Step 6—Turn on Q if memory at RA = hex A1 (this step is just to demonstrate that a full byte of data can be entered for the program to examine).

Step 7—Debounce time delay. Hex 20 is loaded into register 6.1. The register is decremented until R6.0 = 00.

```
ADDRESS   CODE   STEP   COMMENTS

0200      630F   1      V3=0F
  02      6400          V4=00
  04      6600          V6=00
  06      6700          V7=00
  08      A300          I=300
  0A      F00A   2      V0=hex key digit (waits for
                        any key pressed)
  0C      0500          Do machine language routine at 0500
  0E      F10A          V1= hex key digit
  10      8011          Let V0=V0 logically ORed with V1
  12      F055          Store V0 in memory at I, I+1
  14      7401          V4+01
  16      3404          Skip if V4=04
  18      120A          Go to 20A
  1A      121C          Go to 21C (no-op)
  1C      A300   3      I=300
  1E      F265          Let V0,V1,V2= memory at I,I+1,I+2
  20      650A   4      V5=0A
  22      F029          Let I=display pattern for LSD of V0 (LSDP)
  24      D565          Show at V5,V6 coordinate
  26      0540          Do machine language routine at 0540
  28      6505          V5=05
  2A      F029          Let I=V0 (LSDP)
  2C      D565          Show at V5,V6
  2E      6514   5      V5=14
  30      F129          I=V1 (LSDP)
  32      D565          Show at V5,V6
  34      054D          Do mach. lang. routine at 054D
  36      650F          V5=0F
  38      F129          I=V1 (LSDP)
  3A      D565          Show at V5,V6
  3C      7701   6      V7+01
  3E      4701          Skip if V7 not 01
  40      1258          Go to 258
  42      4702          Skip if V7 not 02
  44      125E          Go to 25E
  46      6500   7      V5=00
  48      F229          I=V2 (LSDP)
  4A      D565          Show at V5,V6
  4C      E39E   8      Skip if V3=hex key LSD
  4E      124C          Go to 24C
  50      E3A1          Skip if V3 not equal hex key LSD
  52      1250          Go to 250
  54      00E0          erase display
  56      1200          Go to 200
  58      7606   9      V6+06
  5A      A302          I=302
  5C      121E          Go to 21E
  5E      7606   10     V6+06
  60      050D          Do mach. lang. routine at 050D
  62      A302          I=302
  64      121E          Go to 21E
```

*Listing 3.*



*The screen with Listing 2 running. The circled area shows the hex character F1, which was entered from the keyboard.*

Step 8—Program is at this step because an EF3 = 1 occurred. This is the right half of the byte being entered. Take memory at R5 and OR it with memory at RA to make up a full byte. Also decrement flip-flop register C.

One additional note on the Q latch: It is turned off in the refresh routine. This program will cause the Q audible tone to be long on the first half of the byte and short on the second half. This makes it convenient to be able to stay in step when entering data.

At this point you might say, "Now I have entered a byte of data. If it equals A1, the Q light will come on with a beep. So what?" That might be a valid question, but now we have laid the groundwork for other machine-language programs to build on.

We now have a subroutine that could be used with any Elf-2 program, for instance. We are now able to input variables to a program when needed. If you look carefully at position 00E0 on the screen, you will see the character entered. This is one way of outputting data, although I must admit it is a bit crude.

A more sophisticated method of inputting and outputting data is to use a combination of CHIP-8 language with machine-language subroutines. The CHIP-8 language combined with the display patterns stored in the 512 byte ROM do a nice job of displaying numbers on the screen. The programs in Listings 3 and 4 (the CHIP-8 and machine language, respectively) show the two languages combined. These programs use CHIP-8 for input and output, with machine-language shift and addition subroutines.

The main purpose of these programs is to demonstrate the technique of interfacing CHIP-8 to machine language when there is a desired function that cannot be programmed in CHIP-8 alone. It is also useful if you are new at working the hexadecimal.

Try keying in some hex numbers and observe how hex addition works. The addition routine is an adaption of the routine published in *Kilobaud Microcomputing*, March 1979 ("Programming the 1802," p. 122). That article was written with the Elf-2 in mind. These programs show how Elf-2 programs can be adapted to the VIP. The program in Listing 3 works as follows:

Step 1—Initialize variables.

Step 2—Accept four bytes from the keyboard and store in memory locations 300,301,302, 303.

Step 3—Load memory bytes 300,301,302 in variables 0,1 and

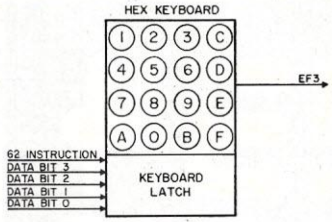



*Fig. 1. Keyboard operation. A 62 instruction causes the least significant four bits of the memory byte pointed to by register X to be latched in the keyboard latch. Only the hex key representing those four bits will generate an EF3 = 1 when pressed. The normal programming technique is to scan each hex digit into the latch and then check for an EF3 = 1 to tell which key has been pressed.*

```
ADDRESS  CODE     STEP  COMMENTS

0500     F806BF   A     RF=06F0
  03     F8F0AF
  06     0FFE           Memory at RF→D, shift
  08     FEFEFE         left  4  times
  0B     5FD4           D→M at RF,return to CHIP-8
  0D     F803BC   D     RC=0300
  10     F800AC
  13     EC             X=C
  14     F800F6         00→D, shift right (clear DF)
  17     4CBD           Memory at 300, operand 1 high
                        order→RD.1, RC+1
  19     4CAD           Memory at 301, operand 1 low
                        order→RD.0, RC+1
  1B     1C             RC+1  (now=303)
  1C     8D             RD.0→D (operand 1 low order)
  1D     F4             Add m at RC +D
  1E     AD             D→RD.0 (low order answer)
  1F     2C             RC-1 (now=302)
  20     9D             RD.1→D (operand 1 high order)
  21     74             Add with carry, m at RC+D,DF
  22     BD             D→RD.1 (high order answer)
  23     5C             D→m at RC, (store high order
                        answer at 0302)
  24     1C8D           RC+1,D.0→D
  26     5C             D→m at RC, (store low order
                        answer at 0303
  27     1C3B2E   E     RC+1 branch if DF=0 (test
                        for carry out of high order)
  2A     F8015C         M at RC (304)=01
  2D     D4             Return to CHIP-8
  2E     F8005C         M at RC (304)=00
  31     D4             Return to CHIP-8
  32     000000
  35     000000         00 up to 540
  38     000000

  40     F806BF   B     RF=06F0
  43     F8F0AF
  46     0FF6           M at RF→D, shift right
  48     F6F6F6         four times
  4B     5FD4           D→m at RF, return to CHIP-8
  4D     F8F1AF   C     RF=06F1
  50     0FF6           M at RF→D, shift right
  52     F6F6F6         four times
  55     5FD4           D→m at RF, return to CHIP-8
```

*Listing 4.*

**Variable Allocation**

| | |
|---|---|
| V 0 | Most significant data buffer and keyboard entry |
| V 1 | Least significant data buffer and keyboard entry |
| V 2 | Carry position data buffer |
| V 3 | Keyboard character for compare |
| V 4 | Counter for keyboard entry |
| V 5 | X display position (horizontal) |
| V 6 | Y display position (vertical) |
| V 7 | Counter for repeat of display routine |

**Memory Allocation**

| | |
|---|---|
| 0300 | Operand 1 high order byte |
| 0301 | Operand 1 low order byte |
| 0302 | Operand 2 high order byte |
| 0303 | Operand 2 low order byte |
| 0304 | Carry byte |
| 0302 | Answer high order byte |
| 0303 | Answer low order byte |

Note: The answer is overlayed into memory locations 0302 and 0303 after operand 1 and 2 have been displayed.

*Table 1.*

2.

Step 4—Set display position. Display variable 0 twice—first the least significant digit, then the most significant digit.

Step 5—Set display position. Display variable 1 twice—first the least significant digit, then the most significant digit.

Step 6—Decision step. First, second or third time at this point. 1st—Have just displayed operand 1, branch to step 9. 2nd—Have just displayed operand 2, branch to step 10. 3rd—Have just displayed answer, go to step 7.

Step 7—Set display position. Display variable 2 (contains carry indication—00 or 01).

Step 8—Check for keyboard = 0F. If key = 0F, wait until it is released. Erase screen. Go to step 1.

Step 9—Set display position and I to display operand 2. Branch to second part of step 3.

Step 10—Set display position. Branch to add subroutine. Set I to display answer. Branch to second part of step 3.

### Machine-Language Steps

Step A—Get variable 0 stored at memory location 06F0. Shift left four times. Put back in 06F0 location.

Step B—Get variable 0 stored at memory location 06F0. Shift right four times. Put back in 06F0 location.

Step C—Get variable 1 stored in memory location 06F1. Shift right four times. Put back in 06F1 location.

Step D—Add subroutine. The program listing is fairly self-explanatory. For a more detailed explanation see "Programming the 1802." This routine is an adaptation of a routine in that article.

Step E—Check for carry (DF = 1). Store 01 for a carry indication at memory location 0304. Store 00 for a no carry indication.

### Conclusion

The program flowchart in Fig. 2 and the memory and variable usage in Table 1 should allow you to follow along with the program. Some shifting is necessary when working with CHIP-8 variables. The keyboard entry and display routines in CHIP-8 will only input and output the least significant half of the byte. The method I used is to go to the memory location the variable is stored in and manipulate it so the least significant half of the byte has the correct data in it. The CHIP-8 program doesn't know the difference.

Well, there you have it. I hope some of these ideas and techniques will be useful for other COSMAC VIP owners. ■

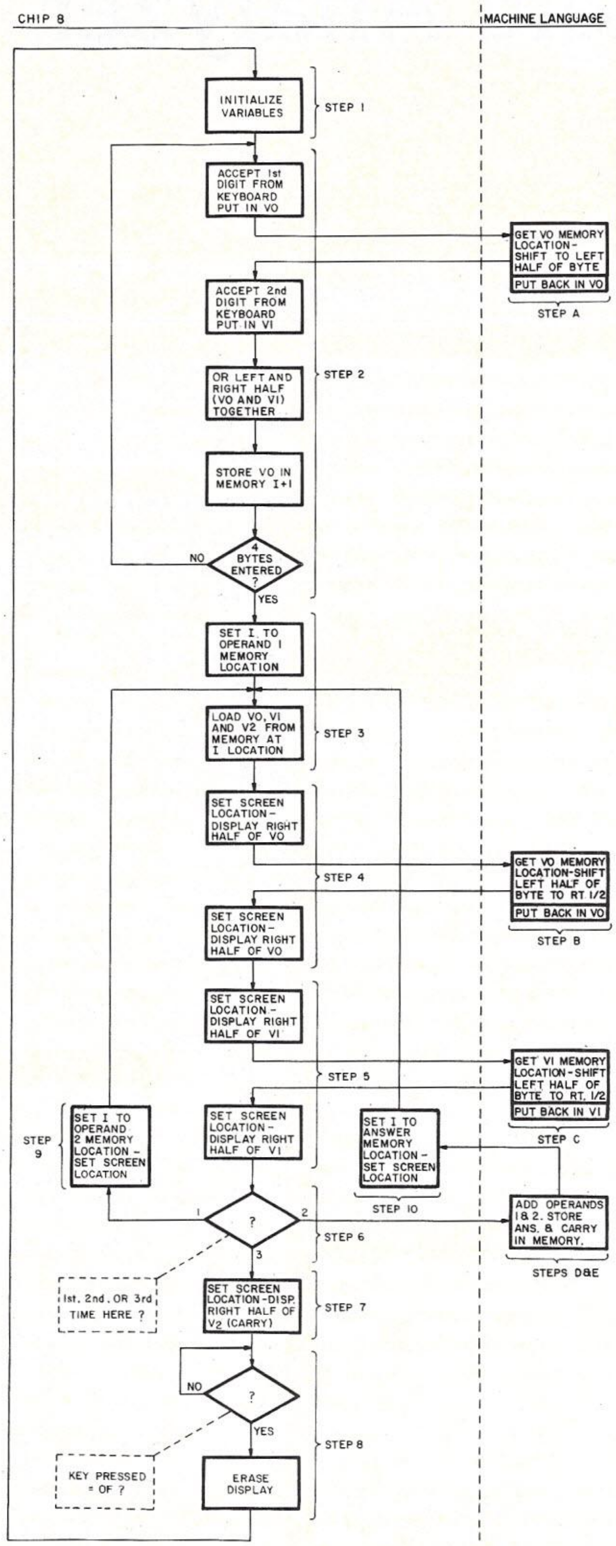



Fig. 2.

# Poor Man's Logic Analyzer

*Troubleshooting needn't be a rich man's sport.*

*Scott B. Eckert*
*113 Roxboro Circle, Apt. 6*
*Syracuse, NY 13211*

When I had completed enough of my home-brew computer system to warrant power on, I was sadly disappointed. It didn't work! My home-brew design is based on the 6502 microprocessor and the 6530-004 multipurpose chip. One of the 6530-004's purposes is to provide a 1K ROM monitor program called TIM (Terminal Interface Monitor). Since it also provides a serial I/O port, I had no front panel controls except for power and reset. Well, I tried reset and got no response—so I was in trouble.

### Background

Let me explain how TIM communicates with the outside world. When reset is pushed, the 6502 processor wants to go to locations FFFC/FFFD to get the two bytes present there. It then interprets these two bytes as a sixteen bit address to which to go and begin executing code as part of the reset initialization program for the particular system in which it is used.

However, logic on my 6530-004 board detects this reset and FFFC/FFFD condition and makes the processor fetch the bytes from 73FC/73FD instead. These two bytes are in the end of the ROM (TIM is required to reside in locations 7000-73FF) and point to the reset routine that is in the lower part of the ROM.

This reset program does the housekeeping chores required for proper system operation, i.e., it initializes the I/O port, initializes the interrupt vectors, sets up the stack pointer, etc. In addition, this routine waits for a carriage return from the keyboard and determines the baud rate at which the serial data is being transmitted.

Anytime you wish to change baud rates, you simply press reset and hit a carriage return. The routine stores this baud rate constant for use whenever an output routine is called. After all this and more, TIM is nice enough to print all the register contents and wait for a command (when it is working properly).

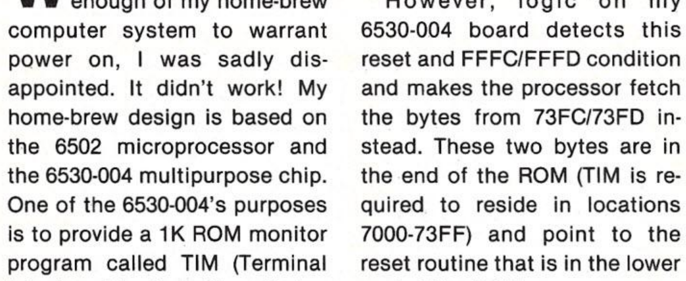



*Fig. 1. Clock signals and various important times on a 6502 microprocessor.*

### The Problem

I knew the processor was doing something in ROM because the chip-select line on the chip was being enabled, but I had no idea what the processor was doing or where it was in ROM. From the address lines, I could determine that it was looping somewhere in the ROM. I had a listing of the TIM program, but that didn't help me at this stage of the game. It is difficult to decipher sixteen address lines simultaneously with a single trace, "trigger-sometimes" scope. Worse, I had to know if the correct data was being sent down the data bus from the ROM itself.

Well, I wished at that point that I knew somebody with one of those fancy logic analyzers. . . you know, the ones that cost hundreds to thousands of dollars! I then decided to build the poor man's logic analyzer—something that would allow me to specify the address that I want it to, grab the data off the data bus and display it to me out in the real world. As you will see shortly, this turns out to be a fairly easy task, really nothing more than a souped-up parallel I/O port!

### The Circuit Details

The DIP switches set the address desired to be the trigger address. Each group of four switches goes to the input of one of the 74LS85 4-bit magnitude comparators. The address line corresponding to each switch is also tied to the proper input on the 74LS85. When the proper address is on the address bus (the address that matches that set into the DIP switches), the A = B output of each comparator drives the cascading A = B input of the following comparator. If all are enabled, the output from U4-6 will enable the 3 input NAND gate U6-2.

If the arm switch has been pressed and flip-flop U5 is set, the rising edge of the Ø2 clock pulse will cause the output of U6A-12 to go low until one of the inputs to it goes low again. Since the address in a 6502 system is valid shortly after Ø1, the level on U6-2 is high well before Ø2 comes along (see Fig. 1).

When Ø2 goes high, the negative-going signal from U6A-12 is inverted again by U6B and used to enable the 7475 quad-latches. The negative-going edge at pin 12 of U6A also triggers U10, a one-shot multivibrator with approximately a 1

second pulse width. This drives an LED to show that the circuit has been triggered by the occurrence of the address set into the switches.

When Ø2 returns low again, pin 12 of U6A goes high. This positive-going edge toggles flip-flop U5, if the analyzer is in the single-shot mode, or does not toggle it if in the continuous mode of operation. In the single-shot mode, the circuit is locked out from further triggers until the arm switch is momentarily moved to the continuous position in order to set the flip-flop. When set to the continuous mode, it can be used as a standard latched parallel I/O port with LEDs, but it can also be easily moved to any address in memory.

### Some Timing Details

I think a word is in order here regarding the timing considerations of this circuit to help adapt this circuit to non-6502 systems. First, the address becomes valid about 200 nsec after Ø1 goes high. The data bus is floated during this period on a 6502, so there is nothing on the data bus to grab yet.

Data becomes valid approximately 150-200 nsec after Ø2 goes high for a write cycle and must be valid for a minimum of 100 nsec before Ø2 goes low for a read cycle. These times are given for a 1 MHz clock, each phase being about 500 nsec. This allows plenty of time for the data to be at the 7475's inputs and meet the data set-up time of 20 nsec. (Data set-up time is defined as the time required for the data to be present at the input terminals before trying to latch it in.)

When the Ø2 clock goes low, this is the most critical part of the whole operation. The data from a 6502 is valid for approximately 10-30 nsec after the Ø2 clock phase goes low. With the two gate delays from the 74LS10 totaling 20-30 nsec, it appears questionable whether the circuit will work. However, the typical delay of a 74LS10 is about 10 nsec, and the typical valid data time is 30 nsec, so every 74LS10 that I tried worked perfectly.

I suppose this stems partly from the fact that most devices are rated conservatively, and, of course, assuming you don't buy junk. You could always go to a 74H10 version of the gate, which has only a 6 nsec typical gate

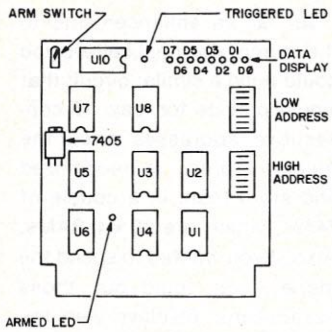



*Fig. 3. Integrated circuit and LED layout. Resistors were placed on end between IC packages to conserve space.*

| Part | Qty. | Price ea. | Total |
|---|---|---|---|
| Radio Shack Board # 276-154 | 1 | $3.50 | $3.50 |
| 74LS85 | 4 | 1.00 | 4.00 |
| 7474 | 1 | .35 | .35 |
| 7475 | 2 | .50 | 1.00 |
| 74LS10 | 1 | .25 | .25 |
| 74121 | 1 | .35 | .35 |
| 7805 | 1 | 1.25 | 1.25 |
| 8-pos DIP switch | 2 | 2.25 | 4.50 |
| LED | 10 | .20 | 2.00 |
| 10k Resistor ¼ W | 17 | .05 | .85 |
| 1k Resistor ¼ W | 10 | .05 | .50 |
| 100 uF 16 V Elect | 1 | .20 | .20 |
| Switch, Toggle SPST | 1 | 2.00 | 2.00 |
| .01 Disk caps | 4 | .05 | .20 |
| | | | $20.95 |

*Table 1. Parts list. The entire circuit was constructed on a Radio Shack experimenter board with the standard 44-pin edge connector. The total price can be reduced if you have a well-stocked junk box.*

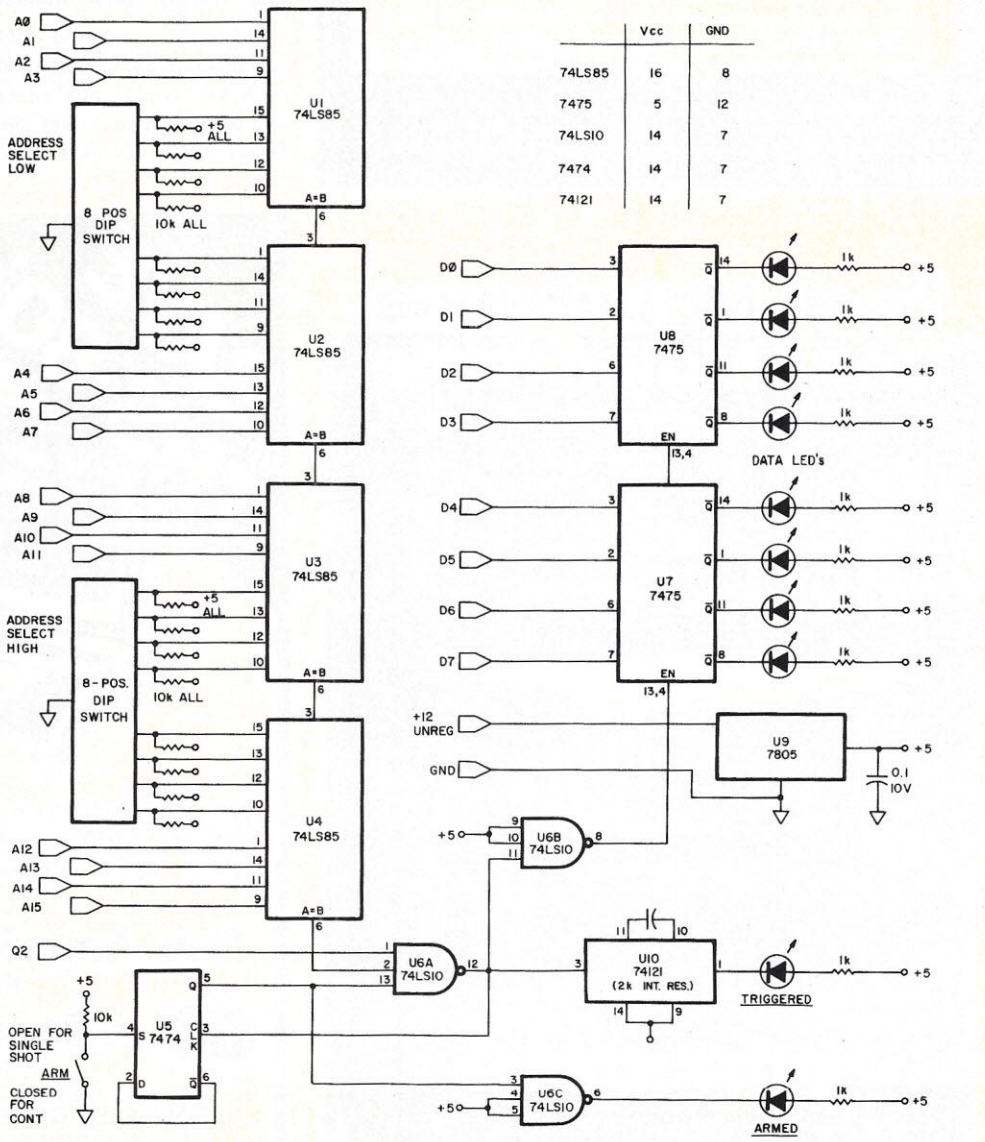



*Fig. 2. Logic diagram of the poor man's logic analyzer.*

delay. I did not try one, but that would probably guarantee success (if that's possible), although I doubt you'll have any trouble with the circuit as shown.

### For My Next Project

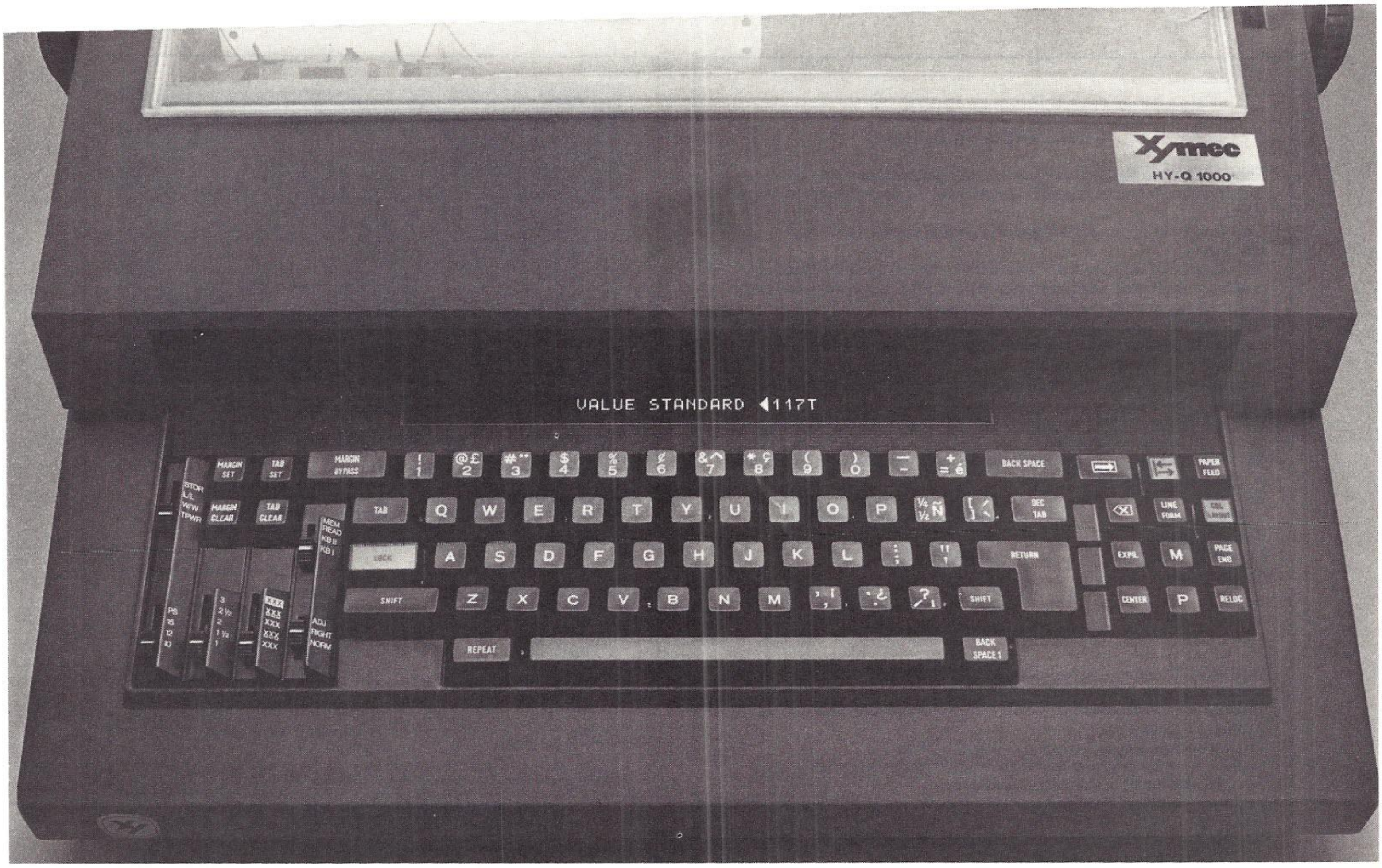
As far as enhancements to the current design, I believe you could build a similar circuit that would provide for, say, 16 consecutive addresses once the trigger address is recognized and store them in a couple of 7489s, which are 16×4 RAMs. Also, if you wanted to spend the money, you could buy those hexadecimal displays with the latch and decoding all in one package. These would make for a nice display, but would increase the cost of the project significantly.

### Did I Get My System Running?

If you are wondering whether or not I got my system running with the help of this analyzer, the answer is yes! I set the address trigger to an address in the section of the monitor that was supposed to run each time a reset was pressed. I checked that the program actually passed through that point, as well as if the data from the ROM was correct.

I continued this until I found the problem area to be the baud rate measuring routine. I checked the baud rate generator board I had built and found it was putting out a 1200 baud rate even though I had it set for 600 baud. This was evidently too high for the TIM program to properly read. I corrected the problem on the baud rate board, set it to 600 baud, pressed reset, and it printed all my register contents and sat waiting for my command.

I can attribute my success to my poor man's logic analyzer. I'm sure I would have found the problem eventually, but I'm also sure it saved me a lot of valuable time. . . much better spent trying to devise ways to improve my system, not merely getting it to work! ■




# Now...
# Intelligence beyond your CRT

## HY-Q 1000

Xymec's HY-Q 1000™ Series brings microprocessor capability to printing. This fast, letter quality, intelligent printer/typewriter expands the capability of your computer or word processing system.

**Xymec gives your system more of the features you want most:**

- **L.E.D. display**
- **tractor feed**
- **RS232, IEEE-C779 interface**

Compatible with nearly all microcomputers, the HY-Q 1000™ Series prints **boldface** and reverse, centers titles, sets columns and right justifies. Your choice of three pitches and proportional spacing, with minimum throughput print speed of 20 cps. Changing print wheels or type styles takes but seconds.

**Intelligent printing makes good sense to me:** Send me more information and the name of my local dealer.

Name ____________________

Title ____________________

Company ____________________

Address ____________________

City ____________________

State ____________ Zip ____________

Telephone ____________________

Xymec, Inc.
17905 Sky Park Circle, Suite J
Irvine, CA 92714

**Xymec™**

**The Value Standard in Printers**

Xymec gives you more features than any other machine on the market, regardless of price!

Xymec intelligent printers won't create a service nightmare. Reliably constructed, the HY-Q 1000™ Series is built with serviceability in mind. And with service centers nationwide, maintenance is easy and convenient.

Give your computer system the intelligent edge. Join up with Xymec. For more information, send in this coupon today.

Xymec, Inc.
Subsidiary of Litronic Industries
17905 Sky Park Circle, Suite J
Irvine, CA 92714 (714) 557-8501

## Client Write-Up System*

The Client Write-Up System allows for the quick preparation of financial records (General Ledger, Balance sheet, Income Statement) for a client.
One time entry of general ledger and payroll accumulation by employee.
Calculates and accrues employer taxes for Social Security, Federal and State U/C taxes. User furnishes the taxable wages and rates and the computer sets up the liability and expense.
Allows up to ten repetitive entries for depreciation and amortization of prepaid expenses, which are entered into the Client Master File and automatically updated monthly.
General ledger account and employee names are displayed when keyed by the user.
Can consolidate individual master files into one profit and loss statement, and one balance sheet.
All constants such as tax rates, taxable wages, etc. are user furnished in the Client Master file.

**Reports**

| | |
|---|---|
| Trial Balance | Comparative I/S and B/S |
| General Ledger | 941, W-2 |
| Income Statement | $1500.00 |
| Balance Sheet | |

## Time Analysis System*

Accurate accounting of the billing of time and services rendered is the goal of the Time Analysis System (TAS).
Activity file can be custom tailored by the organization to fit individual business applications.
Employee time is categorized into different activities.
Activities are entered into the client file and reflect the activity performed, and the time spent by the employee.
Rates are assigned to each employee and are used to bill the employee's time.
Rate and activity amounts to be billed are computed and maintained as a Work-In-Progress amount.
All or part of the Work-In-Progress amount is transferred into an accounts receivable amount at the end of the month.
Statements are printed reflecting the accounts receivable amount owed by the clients.

**Reports**

| | |
|---|---|
| Activity File | Client Activity Ledger |
| Employee File | Statements |
| Employee Activity Report | $200.00 |
| Client File | |

## Asset Depreciation System*

The goal of the Asset Depreciation System (ADS) is to keep accurate records of a client's assets and subsequent depreciations.
Depreciation of a given asset is computed by one of five standard methods: Straight Line, Sum-of-the-Year's-Digits, Double Declining Balance, 1.50 Declining Balance, and 1.25 Declining Balance.
Dates used with the purchase of the asset are rounded to the nearest month start.
An asset, at the end of useful life, is terminated by one of four methods of disposal.
ADS houses a general ledger chart of accounts so that the accountant's own general ledger may be easily cross-referenced with the ADS.
The asset cost and accumulated depreciations are grouped by the ADS general ledger.

**Reports**

| | |
|---|---|
| Fixed Asset Ledger | COA listing |
| Asset Depreciation Schedule | Equipment File Listing |
| | $200.00 |

| | |
|---|---|
| ***Hardware Requirements** | or |
| MOD IIw/ 64K RAM | MOD I w/ 32K RAM |
| 132 Column Line Printer | 3 5¼" Disk Drives |

**HMCT The Complete Computer Company** ✓33

Houston Micro-Computer Technologies, Inc.
5313 Bissonnet Bellaire, Texas 77401 713/661-2205

# Have computer, will travel.

## Executive Computer System Carrying Cases.

- Makes your microcomputer truly portable.
- Protects your equipment: locking latches limit access.
- Rugged black vinyl with metal corners outside.
- Protective foam rubber, black velveteen covered, inside.
- Computer can be operated without removing from case.
- And cases are custom designed for full systems.

**Apple* Executive Case holds:**

- Apple microcomputer.
- 9" Sanyo monitor.
- 2 disk drives.
- Power strip.
- 2 boxes diskettes.
- Manuals.
- Dimensions: 28" x 21" x 10½"
- Weight: 17 pounds.
- Price: $179

**TRS-80** Executive Case holds:**

- TRS-80 Microcomputer.
- Expansion interface.
- 2 disk drives.
- Power strip.
- 2 boxes diskettes.
- Manuals.
- Dimensions: 28" x 21½" x 8½"
- Weight: 17 pounds.
- Price: $179

Terms: FOB Los Angeles—Master Charge, Visa or check with order. Allow 3-4 weeks for delivery.

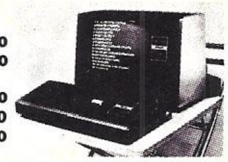
*Registered, Apple Computers, Inc.
**Registered Trademark, Tandy Corporation.

**COMPUTER TEXTile**

10960 Wilshire Blvd, Suite 1504 ✓119
Los Angeles, CA 90024 (213) 477-2196

"TRS-80 is a registered trademark of TANDY CORP."

**Call about our fantastic price on 4-drive complete system.**



| | |
|---|---|
| Level II 4K | $557.10 |
| Level II 16K | $720.00 |
| Expansion Interface | $269.00 |
| Expansion Interface 16K | $403.20 |
| Expansion Interface 32K | $524.00 |
| 16K Memory Kit for TRS-80 or Apple | $79.95 |

**TRS-80 & NORTH STAR ADD-ON DRIVES**

**CUSTOM ENCLOSURE**

**CABLE INCLUDED**

| | |
|---|---|
| Single drive system in custom enclosure | $400.00 |
| Single drive system in metal enclosure | $375.00 |
| Double drive system in custom enclosure | $806.00 |
| MPI | $275.00 |
| MPI, B52, dual headed | $349.00 |
| Shugart SA400 | $275.00 |
| Shugart SA800 | $479.00 |
| Tandom single sided | $279.00 |
| Tandom double sided | $425.00 |
| Hazeltine 1000 | $450.00 |
| Single tier walnut enclosure for Shugart | $ 35.00 |
| Double tier walnut enclosure for Shugart | $ 52.00 |
| Atari 400 | $548.49 |
| Atari 800 | $795.00 |
| Hazeltine 1410 | $749.00 |
| Centronics PI Printer (TRS-80 add on) | $398.95 |
| Centronics 779-2 tractor (TRS-80 add on) | $1049.95 |
| TI Printer | $1599.00 |
| Base 2 | $649.00 |
| Horizon 1, 32K | $2290.00 |
| Televideo 912 | $775.00 |

**SPECIAL! MINI FLOPPY DISKS, box of 10 (with plastic box) only $28.00 (without plastic box) only $26.50. Box of 10. 8" disks (in plastic box $30.00). Centronic 779 ribbons $3.50 each.**

**Itora systems INC** ✓65

*WE ACCEPT BANK AMERICARD, VISA, MASTER CHARGE*

29-02 23rd Ave., Astoria, N.Y. 11105
TWX 7105822107 (212) 728-5252
(800) 221-1340

**Dear Subscriber:**

**Kilobaud Microcomputing does not keep the subscription records on the premises, therefore, calling the Peterborough offices doesn't solve your subscription problem.**

**To quickly solve your problem please send your most recent address label and a description of the problem to:**

**Kilobaud Microcomputing**
**Subscription Department**
**PO Box 997**
**Farmingdale, NY 11737**

**Please allow the subscription department at least two weeks for an answer or a solution to your problem.**

**Thank you and enjoy your subscription.**

**Sincerely,**

Debra V. Bord

**Debra L. Boudrieau**
**Circulation Manager**

# A Humanist's Approach To Computer Programming

*You can tell a lot about a person by the programs he writes.*

*Dick Lutz*
*4802 Fifth Avenue*
*Pittsburgh PA 15213*

In some sense, the subtitle of this article might be "A Computerist's Approach to Human Programming." That is, the intention is to program *you*, the writer of computer programs for the use and enjoyment of yourself and others, so that what you prepare to run on your computer will also run effectively in your mind—and in *mine* should I have the occasion to read one of your programs (or worse, attempt to debug it).

"What," you may ask, "is the percentage in this idea?" Why should you write programs so that I can read them? Simply because unless you've never written anything more complicated than a counting program, you've probably had the experience of trying to figure out what in the world you were trying to do in one section or another of one of your programs. Surely you've puzzled over somebody else's poorly annotated program, wondering what in the world he had in mind.

Is this a plea for documentation? In part, yes. It's time to start writing programs for people to read—not just machines. My reasoning is that if you write REMarks good enough to help you understand your program later, there's at least a fighting chance that if I encounter it, I'll eventually be able to untie the knot it'll put in my feeble cranium.

But there's even more to it. As computerists, we're on or near the leading edge of something that will become more and more widespread in our society—the use of electronic computers to lighten the load and enlighten life. That makes it particularly important that we computerists think like humanists in writing programs. . .simply because more and more people are going to be trying to read them.

The exact approach is not important—only that there be a logical (humanly logical, please) consistency to what you're doing, so that others can pick up and read your style. Rather than lecture about how to proceed, let me tell you how I go about it, and you can adapt my procedures to those you've already worked out for yourself.

## PRGMBASE

I keep PRGMBASE handy. It's reproduced here. I load it whenever I begin writing a program in Microsoft BASIC. It not only imposes form on what I write, but it also helps me as I write.

## Program Writing Utilities

In lines 0–9, you'll find some utility items specifically written for my hardware configuration. By typing RUN/cr, I call line 1 into action. It alternately expands or contracts my Imsai VIOC video output. This is convenient because on my fussbudget monitor, the 24 × 80 format occasionally leaves doubt as to whether I'm looking at 0, 8, 2 or 5, all of which can look alike at the screen's edges.

Line 3 toggles video in and out of the wraparound mode, so RUN 3/cr gives me scrolling whenever I need it. The 0–9 portion of PRGMBASE is selected for this duty because it simplifies repeated call-up while writing programs. These utilities could as well be in the 65000 zone, except that it would take more typing ("RUN 65302/cr") for each use.

Of course, once a program is written and I have no further use for these utilities, I just DELETE 0–10 before a final SAVE. But there's more.

## Automatic Variable Index

Ever lose track of a variable? It can be maddening to work for hours to debug a recalcitrant program, only to discover that you've used variable CH as both CHart and as CHange, and maybe even as CHoice. With three potentially overlapping assignments, your program is sure to stumble on a variable sooner or later. I've made this mistake so often that I now keep track of my variables as I go along, using the variables index built into lines 7000–9700 of PRGMBASE.

It's very easy, once you get used to it. A variety of methods are possible, but what works for me is to take a hard-copy listing of lines 7000–9700 as I begin. Then I can see at a glance that the K variables are indexed at 8100, and that variable KS would be indexed at 8100 + 19, S being the 19th letter of the alphabet. Since I don't remember that S is letter #19, line 8900 reminds me.

When I'm about to use a new variable, I determine what its number would be in the index—but before I write the index line for it, I LIST it. If it comes up blank, I know I'm in the clear. If not, I've saved myself some excruciating debugging. And through the process of saving the variables in this index, I've helped myself and others to decipher what my program is supposed to be doing.

If it's carefully planned, the variable index itself can be used to initialize variables, set default values and to DIMension arrays (see lines 8717, 8726 for examples). And as a final bonus, this simplifies the DEFINT process at the end of program writing, since a glance at a section will tell you immediately if all R (or S, or B) variables are integers.

Incidentally, you'll notice in the index that LN is a reserved variable, which means LiNe (number). When I'm writing a program section that later might give me trouble—or if I'm fretting over one that is already giving me fits—I just insert an extra line: LN = (current line number). Then I add the following line to my utilities in the 0–9 section:

```
4 PRINT "BUGSPRAY NEEDED LINE"; LN;
:RETURN
```

By inserting GOSUB4 where I suspect a program is running amok, I get a readout of where my problem may be. Again, there is a variety of approaches to this idea, and you can impose your own method to make it work best for you.

### Program Index

PRGMBASE also provides a program index, which simplifies the work of somebody trying to figure out what the program is intended to do, as well as helping me organize myself while writing a program. So that I don't spend a lot of time figuring out where to index a program section, there's a simple formula in the program for deriving the index location from the program line number, or vice versa:

INDEX – (PROGRAMLINE/100) + 10000

or, in human terms, drop the last two digits from the program line number and add 10 to the left of what remains. Program line 25500 becomes index location 10255. In reverse, it's just a matter of dropping the 10 from the head of the index number and adding 00 at the end of the result. Thus, index location 10025 refers to program line (0)2500.

### Program Structure

In a further sense, the way I write programs in itself amounts to an index. That kind of program structuring is shown in PRGMBASE in lines 10860 and up. Once more, this is not just useful to the poor schlemiel who may have to figure out what I've written, but to me while I'm writing a program—particularly when I must leave it for a few days and want to return with some notion of what I was doing.

I use four (sometimes five) routine levels:

| | |
|---|---|
| DEITY | rarely, in lines 0-9 |
| REGAL | in the 10 to 99 LNs |
| EXECUTIVE | in the 3-digit lines |
| MANAGERIAL | in the 4-digit lines |
| WORKER | in the 5-digit lines |

As a rule (but not without

This program was written in Microsoft disk BASIC version 4.51; in the current versions, 5.0, a space is required between GOSUB or GOTO and the line number. That is, GOSUB9980 is illegal; use GOSUB 9980 instead.

*Program 1.*

```
0 'later will jump into program
1 PRINT CHR$(27);CHR$(67);
       CHR$(27);CHR$(76);
       CHR$(26); :  STOP'   video size
3 PRINT CHR$(27); CHR$(83);:
       STOP' scroll/wraparound switch
5 PRINT CHR$(26);'       clear screen
9 STOP'           backstops utilities 0-9
10 '
98 STOP'          final program stop
99 '
1000 '-----
6999 '
7000 'VARIABLES INDEX
7001 'Var        TYPE    function
7100 'A..01......................A..01..
7200 'B..02......................B..02..
7300 'C..03......................C..03..
7400 'D..04......................D..04..
7500 'E..05......................E..05..
7600 'F..06......................F..06..
7700 'G..07......................G..07..
7800 'H..08......................H..08..
7900 'I..09......................I..09..
8000 'J..10......................J..10..
8100 'K..11......................K..11..
8200 'L..12......................L..12..
8214 'LN          int     LiNe number
8300 'M..13......................M..13..
8400 'N..14......................N..14..
8500 'O..15......................O..15..
8600 'P..16......................P..16..
8700 'Q..17......................Q..17..
8713 'QM          i       Quote Manip'
8717 DIM QQ(5,5)'         Ques array
8726 QZ=1'        i       1=on, 0=off
8751 'Q1          .###    var Quotient
8800 'R..18......................R..18..
8900 'S..19......................S..19..
9000 'T..20......................T..20..
9100 'U..21......................U..21..
9200 'V..22......................V..22..
9300 'W..23......................W..23..
9400 'X..24......................X..24..
9500 'Y..25......................Y..25..
9600 'Z..26......................Z..26..
9697 RETURN'from getting these values
9699 '
9700 'LINE LLISTER UTILITY
9702 PRINT CHR$(26)'     clear screen
9703 CC=32:LN=1:CL=3:GOSUB9980'
       With CC an ASCII blank (32),
       call 9980 (position & print)
       to place cursor at LN1, CL3,
       to display:
9706 PRINT "NEXT >  RUN ¦ to LList"
9709 CL=10:LN=2:GOSUB9980'pos to pnt:
9710 PRINT "9770 ¦ Program Index"
9711 LN=3:GOSUB9980
9712 PRINT "9790 ¦ VARs Index"
9713 LN=4:GOSUB9980
9714 PRINT "9810 ¦ DEITY      0-9"
9715 LN=5:GOSUB9980
9716 PRINT "9830 ¦ REGAL     10-99"
9717 LN=6:GOSUB9980
9718 PRINT "9850 ¦ EXECs   100-999"
9719 LN=7:GOSUB9980
9720 PRINT "9870 ¦ MGRs   1000-6999"
9721 LN=8:GOSUB9980
9722 PRINT "9890 ¦ subrs 10660-10999"
9723 LN=9:GOSUB9980
9724 PRINT "9910 ¦ WRKRs 11000-"
9725 LN=10:GOSUB9980
9726 PRINT "9930 ¦ this utility"
9729 PRINT CHR$(30);:STOP'
       Await run command
9769 '
9770 'LList PROGRAM INDEX
9772 LN=2:CL=1:GOSUB9980'
       place cursor and
9774 PRINT "        LISTING:"'
       so that it covers the
       menu item being run
9776 LN=3:GOSUB9980'
       position cursor,
       mark place with:
9778 PRINT "RUN NEXT> "
9779 LN=2:CL=1:GOSUB9980
9782 GOSUB 9990'   skip 3 lines
9783 LPRINT "PROGRAM INDEX LISTING"
9784 LPRINT'              skip 1 line
9785 LN=2:GOSUB9980'
       repark the cursor to spot
       the "Ok," on completion of
       LList, at the RUN NEXT>.
9786 LLIST 9999-10699
9788 STOP
9789 '
       Remaining modules follow
       the same pattern as
       lines 9770-9788; see sub-
       routines at 9980, 9990.

9790 'LIST VARIABLES INDEX
9794 LN=3:CL=1:GOSUB9980
9795 PRINT "        LISTING:"
9796 LN=4:GOSUB9980
9797 PRINT "RUN NEXT> "
9798 LN=3:GOSUB9980
9802 GOSUB 9990
9803 LPRINT "VARIABLES INDEX"
9804 LPRINT
9806 LN=3:GOSUB9980
9807 LLIST 6999-9699
9808 STOP
9809 '
9810 'LIST DEITY LEVEL
9812 LN=4:CL=1:GOSUB9980
9814 PRINT "        LISTING:"
9816 LN=5:GOSUB9980
9818 PRINT "RUN NEXT> "
9820 LN=4:GOSUB9980
9823 GOSUB 9990
9824 LPRINT "DEITY level"
9825 LPRINT
9826 LN=4:GOSUB9980
9827 LLIST 0-9
9828 STOP
9829 '
9830 'LIST REGAL LEVEL
9832 LN=5:CL=1:GOSUB9980
9834 PRINT "        LISTING:"
9836 LN=6:GOSUB9980
9838 PRINT "RUN NEXT> "
9839 LN=5:GOSUB9980
9842 GOSUB 9990
9844 LPRINT "REGAL LEVEL"
9845 LPRINT
9846 LN=5:GOSUB9980
9847 LLIST 10-99
9848 STOP
9849 '
9850 'LIST EXECUTIVE LEVEL
9852 LN=6:CL=1:GOSUB9980
9854 PRINT "        LISTING:"
9856 LN=7:GOSUB9980
9858 PRINT "RUN NEXT> "
9859 LN=6:GOSUB9980
9862 GOSUB 9990
9864 LPRINT "EXECUTIVE LEVEL"
9865 LPRINT
9866 LN=6:GOSUB9980
9867 LLIST 99-999
9868 STOP
9869 '
9870 'LIST MANAGER LEVEL
9872 LN=7:CL=1:GOSUB9980
9874 PRINT "        LISTING:"
9876 LN=8:GOSUB9980
9878 PRINT "RUN NEXT> "
9879 LN=7:GOSUB9980
9882 GOSUB 9990
9884 LPRINT "MANAGER LEVEL"
9886 LPRINT:LN=7:GOSUB9980
9887 LLIST 999-6999
9888 STOP
9889 '
9890 'LIST misc SUBRs in 10660-999
9891 LN=8:CL=1:GOSUB9980
9893 PRINT "        LISTING:"
9895 LN=9:GOSUB9980
9897 PRINT "RUN NEXT> "
9899 LN=8:GOSUB9980
9903 GOSUB 9990
9905 LPRINT "miscellaneous subrs"
9906 LPRINT:LN=8:GOSUB9980
9907 LLIST 10659-10999
9908 STOP
9909 '
9910 'LIST WORKER LEVEL
9911 LN=9:CL=1:GOSUB9980
9913 PRINT "        LISTING:"
9915 LN=10:GOSUB9980
9917 PRINT "RUN NEXT> "
9919 LN=9:GOSUB9980
9921 GOSUB 9990
9923 LPRINT "WORKER LEVEL"
9925 LPRINT:LN=9:GOSUB9980
9927 LLIST 10999-'        to end
9928 STOP
9929 '
9930 'LIST THIS UTILITY
9931 LN=10:CL=1:GOSUB9980
9933 PRINT "        LISTING:"
9935 LN=1:GOSUB9980
9936 GOSUB 9990
9937 LPRINT "LineLLister utility"
9938 LPRINT
9941 LN=2:GOSUB9980
9943 LLIST 9699-9999
9945 GOSUB 9990
9979 '
9980 'POSITION CURSOR to PRINT
9982 PRINT CHR$(27); CHR$(61);
       CHR$(LN+31); CHR$(CL+31);
       CHR$(CC);'       position cursor
       and print CC (a blank in this
       particular usage).
9984 RETURN
9989 '
9990 'LSKIP 3 LINES subroutine
9992 LPRINT:LPRINT:LPRINT
9994 RETURN
9999 '
10000 ' MASTER PROGRAM INDEX
       to derive an index location for
       a program line, drop the last 2
       digits & add 10000
10001 '          INDEX  2000 at 10020,
                       20000 at 10200,
                       25500 at 10255
10002 ' to turn an index number into a
       prgm line number, drop first 2
       digits & add 2 trailing zeroes.
10003 '            10255 becomes 25500,
                   10021 becomes 02100,
                   10505 becomes 50500
10005 '
10070 '  7000   VARIABLES listing
                A =7100, B =7200,
                A1=7151, B1=7251, etc.
10100 ' 10000   this INDEX
10107 ' 10700+  suggested conventions
10699 '
10700 '-----
10709 '
10730 'SOME SUGGESTED CONVENTIONS
10740 '
10750 'Ln#s     generally
        ending  are used
        in:     for:
10760 '  0      stock entries used with
                PRGMBASE regularly
10761 '  1      draft #1 of a prgm
                (w/ 20-line spacing
                to allow additions)
10772 '  2      draft #2
                (using the alternate
                20-line spacing)
10783 '  3      draft #3 (revisions)
10794 '  4      draft  4
10805 '  5      draft  5
10816 '  6      draft  6
                        NOTE:  As new
                drafts come in, the old
                lines deleted are not
                re-used 'til needed;
                they are simply erased.
10827 '  7      RETURNS
10838 '  8      temporary checklines
                & last ditch stuff;
                treat as "extra" 'til
                desperately needed
10849 '  9      spacer lines
                between prgm sections,
                like 10859 below.
10859 '
10860 'PRGM STRUCTURE conventions
10870 'Levels of operation:
        Deity            1-digit line#s
        Regal            2-digit line#s
        Executive        3-digit line#s
        Managerial       4-digit line#s
        Worker           5-digit line#s
10875 'First digits always deal with
        like matters, gosubbing thru
        the levels making ever-finer
        decisions 'til some worker-lvl
        does actual processing.
10881 'Thus, branching follows this
        pattern, generally:
10885 '    20-   30 REGALS gosub to
          200-  300 EXECs which gosub
to        2000- 3000 MANAG'L, to
        20000-30000 WORKERs, etc.
10891 'RETURNS always "backstitch the
        seam," so 20200-20297 RTNs
        to the 2021+ range.  That could
        have a 2027 RTN to exec, or a
        2030 gosub to 20300.
10901 'Thus the following might be the
        flow of a typical program,
        with > & < indicating gosub
        and return directions.
        [Follow the arrows > right for
        gosubs, left < for returns,
        reading down.]
10902 'The letters A-D indicate
        routine and subroutine levels:
                A = Regal
                B = Executive
                C = Managerial
                D = Worker
10904 '  A       B       C        D
10905 ' 10
        20 >    200 >   2000 >  20000
                                < 20297
                        2030 >  20300
                                < 20397
                        2040
                        2050 >  20500
                      < 2077  < 20797
                210 >   2100 >  21000
              < 217   < 2197  < 21997
        22 >    220
                230 >   2300
                        2400 >  24000
10906 '                           . . .
              < 297   < 2997  < 29997
        30
        ...
        35
        ...
        41 >    410 >   4100
                        4200 >  42000
                                < 42997
                        4300 >  43000
                      < 4397  < 43997
                440
                450
                460 >   4600 >  46000
                                < 46997
                        4700
                        4800 >  48000
10908 '       < 487   < 4897  < 48997
        50
        ...
        60
        ...
        98 final program stop
10999 '
```

exception), the lower-numbered lines are privileged to call the higher-numbered lines as subroutines. This is why the 7000–9700 line numbers were chosen as PRGMBASE's variable indexing section—because, simply, there are no worker-level line numbers to call, at 70000–99000, in most systems running BASIC.

There are exceptions to the calling convention because a particular subroutine may be useful from more than one program level. This approach also means that I can use the first digit of any line number as a clue to its task. All line numbers beginning with 2, for example, might direct introductory remarks to the user and offer choices to be made by him.

In a top-down approach to programming, you'd write the DEITY level first, then the REGAL, and so on. Although it might violate the sensibilities of the hardbit top-down devotee, this method of program organization allows you to write and place a tiny subroutine when a neatly turned approach occurs to you. . .and then to be sure it works even before writing the calling level.

Fig. 1 shows how the calling convention works. It's also shown in PRGMBASE lines 10904 to 10908; if you adapt your own version of PRGMBASE, having this section in it will provide a quickly available reminder until this approach becomes second nature to you.

### LListing Made Easy

A final bonus of this system of program writing is that sticking with the calling conventions means you can selectively produce really useful hard-copy listings. In fact, PRGMBASE contains its own Utility LineLLister in lines 9700–9994. On RUN 9700/cr it generates a video menu that looks like this:

```
NEXT  RUN    to LList              >
      9770   Program Index
      9790   VARs Index
      9810   DEITY 0-9
      9830   REGAL 10-99
      9850   EXECs 100-999
      9870   MGRs 1000-6999
      9890   subrs 10660-10999
      9910   WRKRs 11000-
      9930   this utility
```

The utility then waits for commands with video keeping the place with its pointer, NEXT >, at the next line to run for a top-down listing. Of course, I take the option of listing only what I need for a given process. Using this programming method, that almost always means from one level, say MGRs, through the levels it calls. Also, since lines 400-499 will usually call only sections in the 4000–4999 (and, in turn, 40000–49999) portion, I save myself much confusion in figuring out what to list and what to debug. . .and I cut down the paper I have to haul around, too.

### Objections

As you might suspect, I've discussed this approach to programming with others who spend hours looking at video monitors and poking away at keyboards. Some voice an objection that memory is costly, and they don't have any to spare. My response is that it's getting cheaper, but even now they can use the methods at the start of programming, later deleting excess parts or saving one complete annotated copy of the program and "stripping" another for space considerations. Furthermore, I maintain that PRGMBASE can *save* memory by imposing code-saving outline methods on programmers.

But the most frequent objection surprises me: "But then I can't use RENUMber to bail me out when I run out of line numbers!" Well, yes and no. You can often RENUMber program lines within the strictures of this approach to program writing. Unfortunately, if you have to do an extensive renumbering job on a program, the program index will be nearly useless afterward. But if you think about your programming in a way that will help it all make sense to a (human) program reader later on, you probably will not need the RENUMber option very often.

It's worth the thought, and the time. It is aesthetic to write neat programs for your personal computer which, if appreciated and observed, can lift your interaction with the machine to a new level of elegance. Try it! It'll make your machine just a little more human—not to mention what it'll do for you! ■

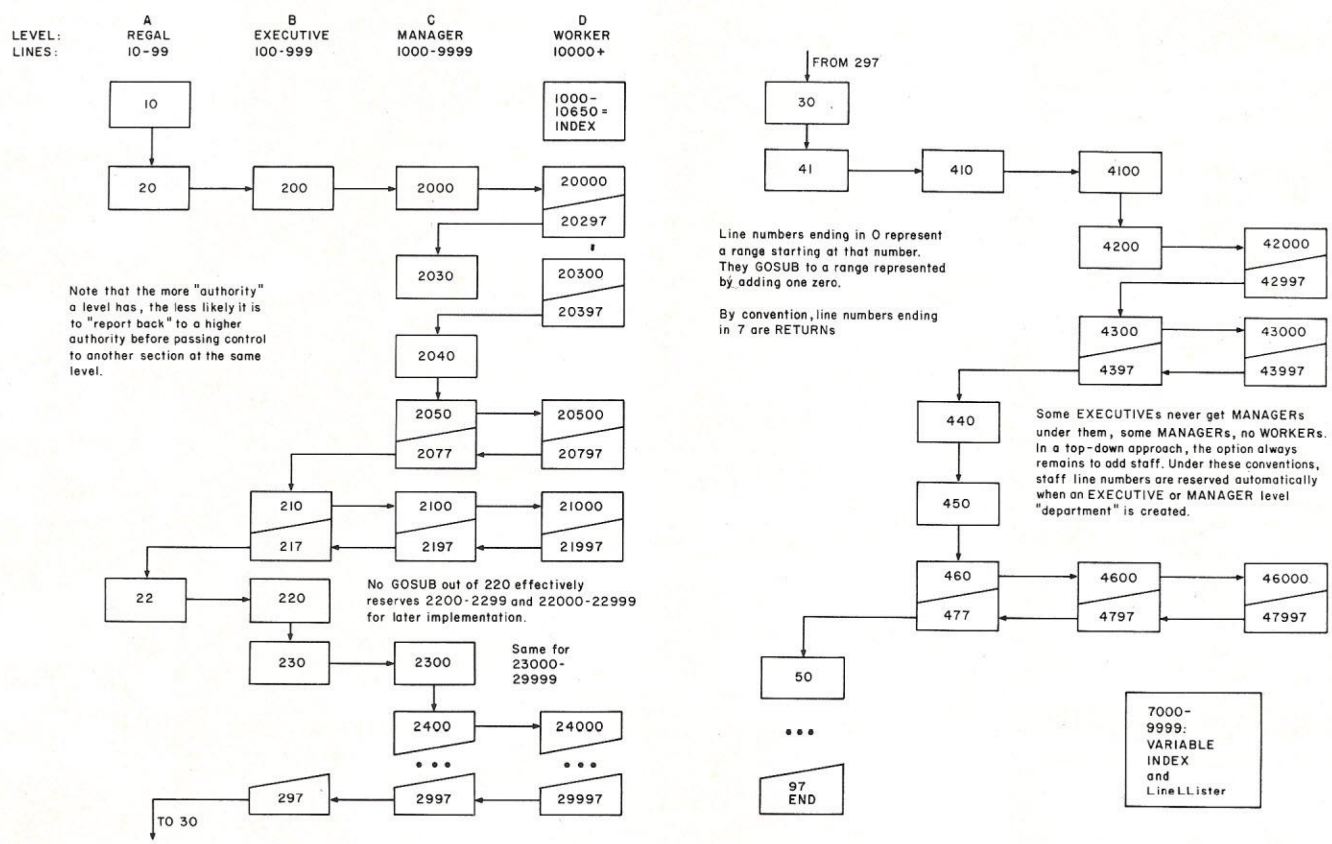
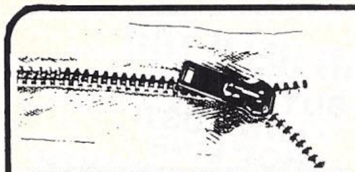



*Fig. 1. Calling convention.*



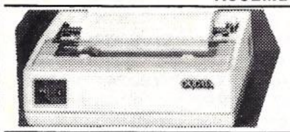
# ZIP UP TO 7 SPEEDS!

**NOW YOU CAN RUN YOUR TRS-80 RELIABLY 100% TO 125% FASTER (4 MHZ)! Our NEW speedup board enables programs to run 50% slower than normal, normal, and 50%, 70%, 90%, 100%, or 125% faster. A 50% minimum increase is guaranteed (90% to 100% typical); however, DETAILED INSTRUCTIONS SHOW CHANGES REQUIRED TO THE TRS-80 THAT WILL INSURE RELIABLE OPERATION AT THE 100% OR 125% INCREASE! Software speed control with switch override option allows speed changes AT ANY TIME without program interruption. Our board also compensates for slow memory! Automatic slow down possible during cassette or disk operation (not required for TRSDOS, NEWDOS, and VTOS 4.0!). Power LED changes color (red - yellow - green) to indicate normal, slow, and high speed operation.**

**ASSEMBLED AND TESTED $37.50**

**VIDEO I. Provides black characters and graphics on an all white screen for a much crisper and easier to read presentation - gives none of the glare associated with plastic screen add-ons. Includes a unit to improve monitor performance.**

**ASSEMBLED $23.95**



**OKIDATA Microline 80 printer.**
**$559 (list $800)**

**Calif. residents add 6% tax. Foreign orders add 10%. Printers shipped freight collect.**

## ARCHBOLD ELECTRONICS

**10708 Segovia Way Rancho Cordova, CA 95670**
**(916) 635-5408**
**Dealer inquiries invited**

VISA

# TOLL FREE ORDERING

**These Fine Products and More**

**NORTHSTAR**
- HRZ 1-32K-D .............2100
- HRZ 2-32K-D .............2340
- HRZ 2-32K-O .............2690
- HARD DISC SYSTEM ......3950

**DYNA BYTE**
- DB 8/1 4 8K ...............2395
- DB 8/2 4 8K ...............3900
- DB 8/4 ....................3030
- 32M PHOENIX ...........11800

**TERMINALS**
- TELEVIDEO 912............745
- TELEVIDEO 920............795
- SOROC IQ-120.............700

**PRINTERS**
- NEC 5510 ................2700
- NEC 5520 ................2975
- TI-820 ...................1650
- ANADEX...................795
- BASE 2 ....................600
- EPSON ..................CALL

**THINKER TOYS**
- DISCUS 2 + 2 1 DRIVE......1265
- DISCUS 2D 1 DRIVE ........970
- DISCUS M26 HARD DISC ...4095

**SOLID STATE MUSIC** Kit Assm
- SB1 SYNTHESIZER ...161 227
- VB1B VIDEO .........125 170
- CB2 Z80 CPU.........168 220

**MEASUREMENT SYSTEMS**
- DM 3200 32K 250ns.........500
- DM 6400 64K 250 ns ........640
- DMB 3200..................650

**SOFTWARE-DISCS-MISC**
- CPM-2.....................150
- WORDSTAR.................350
- GRAHAM-DORIAN .......CALL
- STRUCTURED SYSTEMS ..CALL
- VERBATIM 5 (10) ............28
- VERBATIM 8 (10) ............35
- ATARI ..................CALL
- TI 99-4 ..................CALL

**WE WILL TRY TO BEAT ANY ADVERTISED PRICE**

## Automated Equipment Inc.

4341 W. Commonwealth Ave Suite D
Fullerton, Calif. 92633 ✓96
**(714) 739-4701 (800) 854-6003**

**DISK DRIVE WOES? PRINTER INTERACTION?**
**MEMORY LOSS? ERRATIC OPERATION?**

# DON'T BLAME THE SOFTWARE

Power Line Spikes, Surges & Hash could be the culprit!
Floppies, printers, memory & processor often interact!
Our unique ISOLATORS eliminate equipment interaction
AND curb damaging Power Line Spikes, Surges and Hash.

**ESP Clear up Software and System problems with an ISOLATOR! ESP**

**ALL ISOLATORS: • 125 VAC, Standard 3-prong plug**
- **1875 W MAX Load - 1 KW/Socket or socket bank**
- **Balanced Pi Filtered sockets or socket banks**
- **Spike/Surge Suppression - 1000 Amps, 8/20 usec**

**(SUPER ISOLATORS offer expended filtering and Spike/Surge Suppression capabilities)**

ISO-1

ISO-2

| | | |
|---|---|---|
| ISO-1A | -3 individually filtered sockets .... | $ 56.95 |
| ISO-4 | -6 individually filtered sockets .... | 96.95 |
| ISO-2 | -2 filtered banks; 6 sockets ....... | 56.95 |
| ISO-5 | -3 flitered banks; 9 sockets ....... | 79.95 |

IOS-6

ISO-7

***SWITCHABLE ISOLATORS - ALL ISOLATOR** advantages combined with the versatility, convenience and utility of individually switched sockets. Each switch has associated pilot lite.

| | | |
|---|---|---|
| ISO-6 | -3 switched, filtered sockets ....... | $128.95 |
| ISO-8 | -5 switched, filtered sockets ....... | 161.95 |

***SUPER ISOLATORS** - Cure for severe interference problems. Useful for Industrial applications and heavy duty controlled equipment or peripherals.

- **Dual Balanced Pi Filtered sockets**
- **Spike/Surge Suppression - 2000 Amps, 8/20 usec**

| | | |
|---|---|---|
| ISO-3 | -3 super filtered sockets ......... | $ 85.95 |
| ISO-7 | -5 Super-filtered sockets ......... | 139.95 |

***CIRCUIT BREAKER any model (add-CB) . ADD 7.00**
***CKT BKR/SWITCH/PILOT any model (CBS) ADD 14.00**

**PHONE ORDERS 1-617-655-1532** ✓93

## ESP *Electronic Specialists, Inc.*

**171 South Main Street, Natick, Mass. 01760**

**Dept.** KB-B

# Overlay Programming

*Small memory space? Take a tip from IBM and try this memory-saving technique.*

*Robert A. Peck*
*1276 Riesling Terrace*
*Sunnyvale CA 94087*

Many microcomputer users start out on small systems such as KIM, SYM, COSMAC and D-2. These one-board BASIC computers all have one thing in common: a minimal amount of memory space reserved for the user program. They often allow as few as 256 bytes of RAM space for user access.

Even when we add the remaining on-board memory to the basic unit, we end up with between 512 bytes and 4096 bytes of available space. When a computer user runs out of on-line memory space, regardless of the size of the system being used, he has various choices:

1. Purchase additional memory (with expansion power supplies) and perhaps an adaptation for bus compatibility.
2. Sell off the "basic unit" cheaply to a friend or a broker so you can save money to buy a bigger unit with more memory already installed and rewrite your programs to fit the new processor.
3. Make the programs fit the available memory space by taking advantage of your off-board mass-storage device and the use of an overlay program structure as described below.

Running out of RAM space happens not only to the small micro user, but also to all of the rest of the processor users from small systems to the giants. These larger-system users use a technique that could prove useful to users of minimal systems as well: the overlay structure.

### How the Big-Systems Users Do It

This technique called overlay is often applied to devices in which a functional program takes up almost all of the available memory-storage space. In order to check out the operational characteristics of the device, reserve an area of the RAM for the loading of the diagnostic programs.

If the diagnostic program storage area is small, an interlinked chain of programs may be called in to test each machine function, in turn, and to report the results of that portion of the tests. After this, the next routine in the chain is read into that same memory section as occupied by its predecessor. Then control is given to this next program segment for its diagnostic operation.

It is called an overlay structure because the new diagnostic program entirely erases and replaces the program that precedes it. Each program is stored and executed from only that segment of the memory reserved for the storage of the diagnostic routines.

When the diagnostics are completed, the machine allows the option of either running more diagnostics or returning control to the master program. That master program would, in this case, have taken up most of the available RAM space, and the troubleshooting diagnostics would have taken up relatively little space.

There are several reasons for this:

1. The unit is designed so that it works "perfectly" most of the time.
2. The master program, for economic reasons, takes up most of the space in that there is little reason to install any more memory than absolutely needed for normal operation.
3. The diagnostic routines can be made simple enough so that each may check a small number of basic machine functions.
4. Depending on the capabilities of the mass-storage device associated with it, there may be little reason to limit the total number of diagnostic routines loaded one at a time, except to use sufficient programs to have tested all desired basic functions of the device.

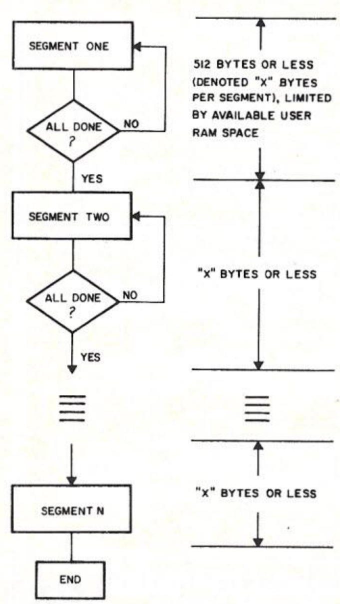



*Fig. 1. The initial step—program segmentation.*

Therefore, for the machine type, which is described above, when power is applied, we would load the master program, which runs all of the basic functional items in the machine, including monitoring the control switches. If a diagnostic routine is desired, that routine is loaded into a separate segment of RAM and is executed. After this, control passes again to the main program, which may load another diagnostic routine or simply continue on with its normal operating sequence.

As stated, the device we examined earlier used most of the RAM space for the storage of the functional program and only a small amount for the diagnostic routines. The small-system user, however, is more likely to reserve a maximum portion of his RAM space for the program material.

### Program Structure

As an initial plan for fitting a large program into a small space, we must first consider structuring the program as noted in Fig. 1. Each of the segments should be capable of fitting individually into the available user RAM space.

Note the linear flow between the segments in Fig. 1. The diagram is intended to show the need for these segments to operate independently. Using the logic outlined in Fig. 2, we can see that a relatively small control program can be used to

```
LDA  NXTSEG  ;load acc with next segment ID number
STA  TAPEID  ;store in tape ID byte search compare area
JSR  TPLOAD  ;call tape-load subroutine (part of monitor)
JMP  STARTX  ;jump to the starting point of this routine
             ;(not the next routine)
```

*Example 1. A typical link call sequence.*

```
0200   Segment 3              0200   Segment 4
        Program                       Program
          --                            --
03F5   LDA 04                 03F5   LDA 05
03F7   STA TAPEID             03F7   STA TAPEID
03FA   JSR TPLOAD             03FA   JSR TPLOAD
03FD   JMP BEGIN03            03FD   JMP BEGIN04
```

*Example 2.*

oversee the loading and execution of a considerably larger segmented program.

Fortunately, many manufacturers of single-board computers write their tape-load programs as subroutines within the monitor program. Although these routines are normally called by the monitor and return control to the monitor, we can use them within our programs and regain control from the subroutines after the tape load has been completed. Thus, we can effectively reduce the size of our control program by efficient use of the monitor subroutines.

There may be occasions, however, when even the relatively small size of the control program tends to get in the way. This could happen if there is a need to transfer a large number of variables between program segments. This forces us to reserve more of the control program space for common area storage.

### Linking Program Segments

If control program space is in a squeeze, we could, instead, place the cassette subroutine call linkage within each of the program segments. Thus, each individual segment would have duplication of the tape-load subroutine call, but we would have possibly reserved more space by this means for the passing of variables between program segments.

As an example, let's look at a typical tape-load routine linkage that might be contained within a single program segment. It could be placed at the logical end of the segment because it is considered the exit point from that segment and the linked entry to the next segment (see Example 1). The last line, at first glance, may be a little difficult to understand, but with a little explanation it is really quite clear.

Since this is an overlay structure, each program segment completely replaces the original segment that occupied that RAM space. When we jump to the tape-load subroutine, the program counter contents are pushed onto the stack. When we complete the tape load, the return address is popped off the stack and reloaded into the program counter.

At this point, the processor executes the instruction currently residing at the location immediately following the locations formerly occupied by the call to the tape-load subroutine. So we went off to a tape-load routine and when we came back to the area originally occupied by the calling program, we were actually jumping into a new program that has since replaced it in that same memory space. Therefore, when we return, we will execute a jump to the start location of the newly loaded segment. By maintaining the same structure in each of the segments, however, we can always assure that the processor knows where to find its next instructions.

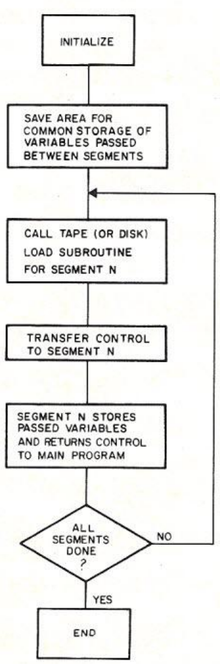



*Fig. 2. The overlay control program structure.*

To demonstrate this, let's put some numbers with the example above (see Example 2). After we have executed the main body of segment 3, we load the accumulator with the value of the next segment's ID code at 03F5. Then at 03F7 we store it in the byte area where the tape-load routine looks for a comparison to the tape ID being read. Now, at 03FA, we jump to the tape-load subroutine from segment 3. This will load segment 4 into locations 0200-03FF, where segment 3 was before.

When we execute the return from the tape-load subroutine, the processor will execute the instruction at 03FD. This instruction, you will note, is now a segment 4 instruction, not segment 3, due to the overlay that has occurred. Therefore, within that segment 4, the instruction indicates a jump to the beginning of segment 4 as other segments will, in this same area, indicate a jump to their own starting areas.

Thus, if all of the program segments are constructed in this manner, we will only be limited by the adaptability of the program to this type of structuring. For the memory-limited programmer, this method may prove to be useful. ■

# A Roundoff Function in Applesoft

*Keep your numbers at a manageable length.*


*Barton M. Bauers, Jr.*
*30 Hillock Drive*
*Wallingford, CT 06492*


When writing programs in Applesoft, you will occasionally need to round off an answer to a specified number of decimal places. Example 1 shows a simple function that will handle almost all of your round-off requirements.

If you place this function at the beginning of your program, you need only to set XQ equal to the number of decimal places you wish to have in your final answer and use the statement VAR = FN RD(VAR), where VAR is the name of any real (floating-point) variable in your program.

### How the Function Works

Review the function and dissect it into its constituent elements. The function breaks down into six pieces:

First, (10 ∧ XQ) raises the value ten to a power that is the same number as the number of decimal places you want in your answer.

Second, X* multiplies the result of (10 ∧ XQ) times the variable you wish rounded.

Third, +.5 adds one-half to this result.

Fourth, INT takes the integer representation of the result of step three.

Fifth, (10 ∧ ( − XQ)) raises the value ten to a power that is the negative equivalent of the number of decimal places you want in the answer.

Sixth, multiply the results of step five by the results of step four.

Consider the example 4.457453. We will use this number to demonstrate how FN RD(X) works. Assume for now that you want this number rounded to two decimal places. Using the six steps above, the function described will calculate the correct value as follows:

XQ will equal 2, the number of decimal places you want in your answer. Therefore, (10 ∧ XQ) will equal (10 ∧ 2), or 10 squared, which is 100.

The variable X is equal to 4.457453. Note that X is only a phantom variable, to be replaced by the actual variable you use. Multiplying X by the result of step one, 100, gives us 445.7453.

To the result of step two we add .5, so our new value is 446.2453.

By taking the integer value of this number, we will get 446.

(10 ∧ ( − 2)) (remember, XQ was set to 2) is the same as 1/(10 ∧ 2), which is equal to 1/100, or .01.

Multiplying 446*.01, we get the final answer: 4.46.

```
DEF FN RD(X) = INT(X * (10 ∧ XQ) + .5) * (10 ∧ ( − XQ))
```

*Example 1.*

### Not Perfectly Round

While this function will provide accurate rounding in almost all of your programs, it is not 100 percent perfect. Some values, when internally represented in your computer, are pre-rounded: the actual value that FN RD(X) sees when it starts its work is not the same number that was keyed in. This will cause a slight under- or over-rounding, but it is not the fault of the function. Indeed, I have worked with FORTRAN programs and have written similar rounding functions and have encountered similar minor rounding differences for some values.

You should therefore carefully consider whether or not an occasional rounding difference is acceptable. If it is not, then you should further question whether or not any rounding is appropriate for that problem. ■

# Accessories and Software

## for Apple, Commodore, TRS-80, Atari

### Call TOLL FREE - 800-421-0347

Hardware for Apple, Commodore, Vector -CALL



apple

## ACCESSORIES

| | |
|---|---|
| CORVUS 10 MEGABYTE DISK DRIVE | $4649 |
| PASCAL LANGUAGE SYSTEM | C |
| GRAPHICS INPUT TABLET | |
| DISK II with CONTROLLER CARD | |
| DISK II without controller | A |
| APPLE SOFT II FIRMWARE CARD | |
| INTEGER FIRMWARE CARD | L |
| PARALLEL INTERFACE CARD | |
| SERIAL INTERFACE CARD | |
| COMMUNICATIONS CARD | L |
| SERIAL INTERFACE CARD | |
| SUP-R-MOD RF TV MODULATOR | 24 |
| SUP-R-TERM 80 col. CARD | 348 |
| DAN PAYMAR Lower case kit | 44 |
| SVA 8" DISK CONTROLLER CARD | 348 |
| CCS ARITHMETIC PROCESSOR CARD | 348 |
| CLOCK/CALENDAR CARD | 238 |
| INTROL X-10 SYSTEM | 238 |
| SUPER TALKER SPEECH SYNTHISIZER | 258 |
| ROMPLUS CARD w/KEYBD. FLTR. | 178 |
| HEURISTICS SPEECHLINK 2000 | 224 |
| DC HAYES MICROMODEM II | 334 |
| ALF MUSIC SYNTHISIZER | 244 |
| SSM A10 CARD (KIT) | 128 |
| SSM A10 CARD (ASSEMBLED) | 169 |
| NOVATION CAT MODE M | 158 |
| CCS GPIB IEEE INTERFACE | 268 |
| MICROSOFT Z-80 SOFT CARD w/CP/M | 348 |
| MICROWORKS DS—65 DIGISECTOR | 348 |
| ROMWRITER | 158 |
| SYMTEC LIGHT PEN CARD | 228 |
| CCS PROGRAMMABLE TIMER MODULE | 158 |
| CENTRONICS PRINTER INT. CARD | 189 |
| SLIENTYPE PRINTER w/INT. CARD | 528 |

## APPLE SOFTWARE

| | |
|---|---|
| 3 MILE ISLAND, Muse | 39.95 |
| 3-D ANIMATION-D, Programma Intl | 24.95 |
| 3D ANIMATION, Sub Logic | 55.00 |
| 3D GRAPHICS, Programma Intl | 24.95 |
| 6502 ASSEMBLER, Personal S/W | 24.95 |
| 8080 SIMULATIONS, Sybex | 19.95 |
| ABBS, Periph Unltd | 49.95 |
| ACCOUNTING ASSISTANT, Instant S/W | 7.95 |
| ADDRESS BOOK, Muse | 49.95 |
| ADVENTURE 3 & 4, Creative Comp | 24.95 |
| ADVENTURE-D, Microsoft | 29.95 |
| ADVENTURE-D, Programma Intl | 21.95 |
| ADVENTURELAND, Creative Comp | 14.95 |
| ALIEN LANDER, Sierra Software | 24.95 |
| AMPERSORT 2, Programma Intl | 15.95 |
| ANALYSIS 1, Galaxy | 49.95 |
| ANALYST, Programma Intl | 15.95 |
| APMAIL, Programma Intl | 19.95 |
| APPILOT, Muse | 49.95 |
| APPLE DOC, Sowest Data | 24.95 |
| APPLE LISNER, Softape | 19.95 |
| APPLE PIE-D, Programm Intl | 24.95 |
| APPLE PILOT, Programma Intl | 24.95 |
| APPLE PLANET, Programma Intl | 24.95 |
| APPLE POST, Apple | 50.00 |
| APPLE WRITER, Apple | 75.00 |
| APPLEFORTH, Programma Intl | 49.95 |
| APPLESOFT OPTIMIZER, Programma | 19.95 |
| APPLESOFT PLUS, Programma Intl | 24.95 |
| APPLETALKER, Softape | 15.95 |
| APPLEWORLD, United Software | 59.95 |
| ASSEM LANG DEVELOPMENT, Hayden | 39.95 |
| ASSEMBLER, Microproducts | 39.95 |
| ASTEROIDS, Quality Softwr | 19.95 |
| ASTROAPPLE, Software Factry | 15.00 |
| AUTOCHECKERS, Programma Intl | 15.95 |
| B I T S, Periph Unltd | 34.95 |
| BABBLE, Software Factry | 15.00 |
| BASEBALL, Muse | 14.95 |
| BATTLESHIP COMMAND, Quality Sftw | 14.95 |
| BATTLESTAR 1, Programma Intl | 15.95 |
| BAZOOKA, Programma Intl | 15.95 |
| BEST OF BISHOP, Softape | 39.95 |
| BLOCKADE, Personal S/W | 14.95 |
| BOWLING, Apple | 15.00 |
| BRAIN GAMES, Creative Comp | 7.95 |
| BRIDGE CHALLENGER, Personal S/W | 14.95 |
| BRIGHT PEN, Softape | 34.95 |
| BUSINESS/FINANCE, Programma Intl | 19.95 |
| CAI PROGRAMS-D, Creative Comp | 14.95 |
| CASTLE ADVENTURE, Programma Intl | 15.95 |
| CCA DATA MGMT., Personal S/W | 100.00 |
| CHECKBOOK, Apple | 20.00 |
| CHECKBOOK, Programma Intl | 34.95 |
| CHECKBOOK KING-C, Personal S/W | 19.95 |
| CHECKER KING-C, Personal S/W | 19.95 |
| CHECKER KING-D, Personal S/W | 24.95 |
| CLOWNS & BALLOONS, Programma | 15.95 |
| COMPUMATH, Edu Ware | 39.95 |
| COMPUREAD, Edu Ware | 24.95 |
| CRAPS, Softape | 14.95 |
| DANS DISK, Periph Unltd | 24.95 |
| DART ROOM, Programma Intl | 15.95 |
| DATA MANAGER, Hayden | 49.95 |
| DATABASE, Programma Intl | 19.95 |
| DATABASE II, Programma Intl | 29.95 |
| DATABASE MAIL-D, Programma Intl | 29.95 |
| DATAMOVER, Apple | 7.50 |
| DDU & SDC, Periph Unltd | 39.95 |
| DEPTH CHARGE, Programma Intl | 15.95 |
| DISASSEMBLER, Microproducts | 34.95 |
| DOW JONES, Apple | 50.00 |
| DUAL RACE, Systems Design | 16.95 |
| EARTHQUEST-D, Programma Intl | 19.95 |
| EASY WRITER, Personal Softwr | 99.95 |
| ELECTRIC CARD FILE, Softape | 19.95 |
| ELECTRIC CRAYON, Muse | 17.95 |
| ELEMENTARY MATH, Muse | 39.95 |
| ENERGY AUDIT-C, Instant S/W | 80.00 |
| ENGINEERING MATH-1, Hayden | 14.95 |
| ESCAPE, Muse | 12.95 |
| FASTGAMMON-C, Quality Softwr | 19.95 |
| FASTGAMMON-D, Quality Softwr | 24.95 |
| FILE HELPER, Baclan | 49.00 |
| FINANCE PACK, Apple | 25.00 |
| FLASH CARDS, Programma Intl | 9.95 |
| FLIGHT SIMULATOR, Sub Logic | 25.00 |
| FOOTBALL, Programma Intl | 9.95 |
| FORM LETTER, Periph Unltd | 24.95 |
| FORTE, Softape | 19.95 |
| FORTH, Softape | 49.95 |
| FRACAS-C, Quality Softwr | 19.95 |
| FRACAS-D, Quality Softwr | 24.95 |
| GALACTIC CONFLICT, Programma Intl | 15.95 |
| GAME PLAY w/BASIC, Hayden | 9.95 |
| GAME PLAY w/BASIC 2, Hayden | 9.95 |
| GAME PLAY w/BASIC 3, Hayden | 9.95 |
| GAMON GAMBLER-C, Personal S/W | 19.95 |
| GAUNTLETS, Programma Intl | 99.95 |
| GENERAL MATH-1, Hayden | 14.95 |
| GLOBAL WAR, Muse | 17.95 |
| GOBLINS, Programma Intl | 15.95 |
| GRADE BOOK, Creative Corp | 24.95 |
| GRAPHIC UTILITIES, Personal S/W | 14.95 |
| HI RES ADVENTURE #1, On Line Syst | 24.95 |
| HIRES PLAYGROUND, Systems Design | 24.95 |
| HOME ACCOUNTING-C, Programma Int | 9.95 |
| HOME ACCOUNTING-D, Programma | 15.95 |
| HOUSEHOLD FINANCE, Programma | 24.95 |
| HOUSEHOLD INVENT., Programma | 24.95 |
| INSTANT LIBRARY-C, Softape | 39.95 |
| INSTANT LIBRARY-D, Softape-39.95 | |
| JOHN'S DEBUGGER, Broderick | 35.00 |
| LASER TURRET, Programma Intl | 9.95 |
| LISA, Programma Intl | 34.95 |
| MAIL LIST, Periph Unltd | 39.95 |
| MATCHWITS, Programma Intl | 9.95 |
| MATH TUTOR 1, Instant S/W | 7.95 |
| MATH TUTOR 2, Instant S/W | 7.95 |
| MATRIX INVERT-C, System Design | 13.95 |
| MAZE GAME, Muse | 12.95 |
| MEMO CALENDAR, Periph Unltd | 29.95 |
| MICRO INFOSYSTEM, Muse | 99.95 |
| MICROCHESS, Personal S/W | 19.95 |
| MICROCHESS-C, Apple | 20.00 |
| MICROCHESS-D, Apple | 25.00 |
| MICROGAMMON, Softape | 14.95 |
| MINI LEDGER, Programma Intl | 59.95 |
| MINI LEDGER-D, Programma Intl | 59.95 |
| MISSION IMPOSSIBLE, Creative Comp | 14.95 |
| MONITOR-D, Programma Intl | 49.95 |
| MORTGAGE, Instant S/W | 7.95 |
| MULTIPLE REGRESSION, Systems D | 24.95 |
| ORIGINAL ADVENTURE-D, Creative Cp | 24.95 |
| OUTDOOR GAMES, Creative Comp | 14.95 |
| P A D, Datasoft | 19.50 |
| P I T S, Periph Unltd | 30.00 |
| P U M P, Periph Unltd | 25.00 |
| PARCHESI, Programma Intl | 6.95 |
| PERCEPTION I,II,III, Edu Ware | 19.95 |
| PERPUTUAL CALENDAR, Programma | 9.95 |
| PHASOR ZAP, Programma Intl | 15.95 |
| PINBALL, Programma Intl | 15.95 |
| PIRATE ADVENTURE, Creative Comp | 14.95 |
| POWER EDITOR, Programma Intl | 15.95 |
| PRO GOLF, Softape | 12.95 |
| PROGRAM EDITOR, Synergistic | 40.00 |
| PROGRAMMING AID, Dakin 5 | 39.95 |
| PROGRAMMING AID 2, Dakin 5 | 49.95 |
| QUARTER HORSE RACE, Programma | 6.95 |
| RAM TEST TAPE, Apple | 7.50 |
| ROCKET PILOT, Softape | 12.95 |
| SARGON 2, Hayden | 27.00 |
| SARGON-C, Hayden | 19.95 |
| SARGON-D, Hayden | 34.95 |
| SCORPION ATTACK, Programma Intl | 15.95 |
| SCREEN MACHINE, Softape | 19.95 |
| SHELL GAMES, Apple | 30.00 |
| SINGLE DISK COPY, Periph Unltd | 19.95 |
| SKEETSHOOT, On Line | 14.95 |
| SOLITAIRE, Programma Intl | 15.95 |
| SPACE, Edu Ware | 29.95 |
| SPACE II, Edu Ware | 24.95 |
| SPACE SHUTTLE LANDING, Harvey | 15.00 |
| SPACE/SPORTS GAMES, Creative Cp | 14.95 |
| SPYTREK, Programma Intl | 15.95 |
| STAR WARS, Softape | 12.95 |

**IF YOU DON'T SEE IT HERE, CALL US. WE CARRY EVERYTHING FOR APPLE, TRS-80, COMMODORE, VECTOR, ATARI, SORCERER, TEXAS INSTRUMENTS, AND MORE!**

| | |
|---|---|
| STATISTICS, Programma Intl | 19.95 |
| STATISTICS, Edu Ware | 17.95 |
| STIMULATING SIMULATIONS, Personal S/W/ | 14.95 |
| STOCK MARKET, RTR | 79.95 |
| SUPER TEXT, Muse | 100.00 |
| SUPERCHIP, Eclectic | 100.00 |
| TALKING CALCULATOR, Softape | 12.95 |
| TALKING DISK, Programma Intl | 19.95 |
| TEXT EDITOR, Periph Unltd | 59.95 |
| THE CASHIER, Apple | 250.00 |
| THE CONTROLLER, Apple | 625.00 |
| THE SOURCE, Periph Unltd | 25.00 |
| THE SOURCE, TCA | 29.95 |
| THE TYPESETTER, Programma Intl | 29.95 |
| TIC TAC TALKER, Softape | 19.95 |
| TIME CLOCK, Programma Intl | 6.95 |
| TIME TREK, Personal S/W | 14.95 |
| TINY PASCAL, Programma Intl | 49.95 |
| TRAPSHOOT, On Line | 19.95 |
| TRIVIA ][, Programma Intl | 15.95 |
| TURF ANALYSIS, Systems Design | 21.95 |
| TYPE TRAINER, Programma Intl | 15.95 |
| TYPING TUTOR, Microsoft | 14.95 |
| UNCLE SAM, Muse | 12.95 |
| UPDATES, Periph Unltd | 39.95 |
| VISICALC, Personal S/W | 150.00 |
| VOYAGER EXCURSION, Programma | 15.95 |
| WEIGHT PLOTTER, Programma Intl | 19.95 |
| WHATSIT, Computerheadgr | 125.00 |
| WINDFALL, Edu Ware | 19.95 |

## PET ACCESSORIES

| | |
|---|---|
| Commodore Dual Floppy Disk Drive | $1,285.00 |
| Second Cassette—from Commodore | 95.00 |
| Commodore PET Service Kit | 30.00 |
| Beeper—Tells when tape is loaded | 24.95 |
| Petunia—Play music from PET | 29.95 |
| Video Buffer—Attach another CRT | 29.95 |
| Combo—Petunia abd Video Buffer | 49.95 |
| TNW Bi-Dir RS-232 printer S-face | 229.00 |
| KIM A (A Single Board Computer from Commodore) | 179.00 |
| CMB 8050 Dual Drive Floppy Disk | 1,695.00 |
| CBM Modem | 395.00 |
| CBM Voice Synthesizer | 395.00 |
| PET TO IEEE Cable | 39.95 |
| IEEE TO IEEE Cable | 49.95 |
| KIM1 & Power Supply Package Special | 200.00 |

## PET SOFTWARE

| | |
|---|---|
| ACTION, Soundware | $9.95 |
| ACTION GAMES, Creative Comp | 7.95 |
| ALPHABET, Commodore | 9.95 |
| AWARI, Programma Intl | 9.95 |
| BACKGAMMON, Commodore | 9.95 |
| BACKGAMMON, Hayden | 10.95 |
| BASIC BASIC, Commodore | 14.95 |
| BASIC FINANCE, Commodore | 24.95 |
| BASIC MATH, Commodore | 29.95 |
| BATTER UP, Hayden | 10.95 |
| BATTLESHIP, Programma Intl | 9.95 |
| BLACKJACK, Commodore | 9.95 |
| BLOCKADE, Programma Intl | 9.95 |
| BLOCKADE, Personal S/W | 14.95 |
| BOWLING, Commodore | 9.95 |
| BRIDGE CHALLENGER, Personal S/W | 14.95 |
| CHASE, Programma Intl | 9.95 |
| CHECKBOOK, Commodore | 24.95 |
| CHECKER KING, Personal S/W | 19.95 |
| CHECKERS, Instant S/W | 7.95 |
| COSTING, Commodore | 19.95 |
| DATA BASE UTILITY, Commodore | 24.95 |
| DEPTH CHARGE, Programma Intl | 9.95 |
| DIET PLAN, Commodore | 14.95 |
| DISASSEMBLER, Commodore | 24.95 |
| DOW JONES, Instant S/W | 7.95 |
| DUNGEON OF DEATH, Instant S/W | 7.95 |
| GALAXY GAMES, Commodore | 9.95 |
| GAME PLAY w/BASIC, Hayden | 9.95 |
| GAME PLAY w/BASIC 2, Hayden | 9.95 |
| GAME PLAY w/BASIC 3, Hayden | 9.95 |
| GRAPHIC GAMES #1, Creative Comp | 7.95 |
| GRAPHIC GAMES #2, Creative Comp | 7.95 |
| GRAPHIC UTILITIES, Personal S/W | 14.95 |
| GRAPHICS, Commodore | 9.95 |
| GUESS A SENTENCE, Commodore | 29.95 |
| HANDS ON BASIC, Commodore | 14.95 |
| HOME ACCOUNTING, Programma Intl | 9.95 |
| LOGIC #1, Creative Comp | 7.95 |
| LOGIC #2, Creative Comp | 7.95 |
| MACHINE LANGUAGE, Commodore | 9.95 |
| MAYDAY, Hayden | 9.95 |
| MICROBASEBALL, Hayden | 9.95 |
| MICROCHESS, Personal S/W | 19.95 |
| MONITOR, Programma Intl | 6.95 |
| MORTGAGE, Instant S/W | 7.95 |
| MORTGAGE, Commodore | 14.95 |
| NUMBER GAMES, Creative Comp | 7.95 |
| NUMBER REVERSAL, Commodore | 9.95 |
| ON THE HOUSE, Commodore | 9.95 |
| PERSONAL ACCT'NG., Programma Intl | 19.95 |
| PET SHOW, Commodore | 9.95 |
| PIRATE ADVENTURE, Creative Comp | 14.95 |
| PROJECTILE MOTION, Commodore | 19.95 |
| REALTY, Commodore | 59.95 |
| RHYMING, Commodore | 9.95 |
| SERIES/PARALLEL, Commodore | 19.95 |
| SLOT MACHINE, Programma Intl | 6.95 |
| SOME COMMON BASIC PROB-D, Osborne & Assoc | 9.95 |
| SPACETALK, Commodore | 9.95 |
| SPACETREK, Commodore | 9.95 |
| STATISTICS, Commodore | 29.95 |
| STOCK PROTFOLIO, Commodore | 24.95 |
| STRATHCLYDE BASIC, Commodore | 14.95 |
| TARGET PONG, Commodore | 9.95 |
| TIME TREK, Personal S/W | 14.95 |
| TREK X, Instant S/W | 7.95 |
| TURF & TARGET, Instant S/W | 7.95 |
| USER PROT COOKBOOK, Commodore | 9.95 |
| UTILITIES, Smithware | 7.95 |
| VIDEO CHECKERS, Compuquote | 14.95 |
| WORD PROCESSING 2, Commodore | 99.95 |
| WORD PROCESSING 3, Commodore | 199.95 |
| WORLD CLOCK, Commodore | 9.95 |

**MICRO BUSINESS WORLD™**

15818 Hawthorne Blvd. ✓248
Lawndale, CA 90260 (213) 371-1660

**CALL US TOLL FREE FOR NEW CATALOG OF ACCESSORIES & SOFTWARE 800-421-0347**

# Clock Control Board

*Speeds up your TRS-80. . . elegantly.*

*Mark A. Schimelman, M.D.*
*255 Townhouse*
*Hershey PA 17033*

The TRS-80 is a great computer . . . almost. The people at Tandy Corporation did a good job of building an efficient, reliable machine at low cost. They did not, however, put everything that they could have into that handsome case to make it the most desirable machine on the market.

Some of the more commonly recognized omissions include the lack of lowercase characters, the lack of standard TV video modulation, the inability to include Level I and II ROMs in the same machine and a relatively slow clock for microprocessor operation.

Fortunately, hardware buffs, being compulsive creatures, cannot stand such a vacuum. They have created a whole library of solutions—some of which are elegant in concept and function—to each of these problems. I recently purchased one such elegant device, the TRS-80 Clock Control Board, produced by William Archbold, 106 Snyder Dr., Mather AFB CA 95655.

I mailed off my $14.95, because Bill promised in his ad that the user could control the clock rate with simple Level II commands and change clock rates without destroying resident programs (a problem I had with previous modifications). My kit arrived promptly eight days afterward and included a clock control board, instruction manual, hookup wire and double-sticky pads for mounting in the computer case.

**Control Board**

The clock control board is a professional-quality 2½ × 1½ inch double-sided PC board with four ICs, a resistor and a capacitor already preassembled. There are nine clearly labeled holes for the connection of the board to the motherboard via jumper wires.

The manual is simple and clearly written with a good diagram showing proper connection points (see Fig. 1) and a schematic of the control board circuit.

Assembly consisted of cutting the hookup wire to appropriate lengths as described in the manual and soldering them first to the clock control board and then to the appropriate points on the motherboard. Finally, a jumper connection was made between Z43, pin Z, and Z56, pin 14. The trace from Z56, pin 8, is cut. Total installation time was one hour, which included double-checking the wiring twice. The board worked perfectly the first time.

When the computer is powered up, the control is automatically set to the normal 1.77 MHz rate. To change the rate to 2.66 MHz, all you have to do is type OUT 254,1 ENTER; to change back to 1.77 MHz, type OUT 254.0 ENTER. This control statement can be part of a program or entered at any time via the keyboard.

As you might imagine, this adds a new measure of flexibility in programming control. Cassette tapes can be read or written at either clock rate, giving effective baud rates of 500 or 750. Computer graphics and computational time can be cut by one-third as well, all under complete computer control. This is truly elegant design and engineering!

Previous clock modifications utilized either mechanical or electronic switches to switch clock rates. The problem was that the switch often occurred at some random point in a microprocessor cycle, leaving the processor in limbo so that by the time the rate settled at a new frequency, the microprocessor was lobotomized. The user then had to power-down and repower-up to sync everything again.

None of that is necessary with the Clock Control Board since it is controlled by an OUT instruction and, hence, always occurs between microprocessor instructions. Thus, you have clock rate control under complete computer control without cutting any more holes in the computer case.

As you may have surmised, I am delighted by the TRS-80 Clock Control Board. It is a simple-to-install, easy-to-understand kit that does what it claims to do. I recommend it highly . . . good job, Bill! ■

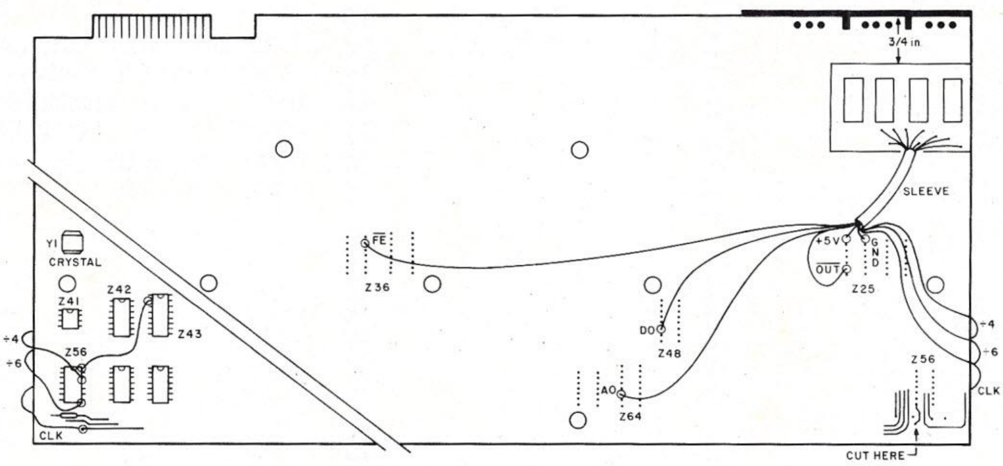



*Fig. 1. Connection points.*

# DEALER DIRECTORY

## El Monte, CA

Ohio Scientific specialist in the San Gabriel valley serving greater Los Angeles. Full product line on display. Specializing in business computers. In-house service. Custom programming. Terminals. Printers. Open Mon-Sat, 9 AM-7 PM. **Computer & Video, 3380 Flair Dr., Suite 207, El Monte, CA 91731, 572-7292.**

## San Jose, CA

Bay area's newest computer store. Featuring the new Texas Instruments TI 99/4 home & business computer. Software for TRS-80, Apple, PET, etc. Magazines. **Hobbi-Tronics, 1378 S. Bascom Ave., San Jose, CA 95128, 998-1103.**

## Santa Barbara, CA

Complete computer systems for business and personal use. Classes, seminars, word processing supplies, books, magazines. **Computers Plus, 1827 State St., Santa Barbara, CA 93101, 963-4542.**

## N. Hollywood, CA

Wholesale prices to dealers & computer club members! Anadex, Atari, Base-2, Centronics, Emako, Godbout, Hazeltine, Lobo, Micropolis, MicroPro, NEC, Okidata, Paper Tiger, Soroc, Tarbell, Televideo, TI, Vector Graphic, Zenith & others. **Patio Computer Sales Co., 5437 Laurel Canyon Bl., #208, N. Hollywood, CA 91607, 762-0020.**

## Sarasota, FL

Dynabyte computer systems, Hazeltine and NEC, Word-Star, Structured Systems accounting. Consulting, training, sales, service. **Glisco, Inc., 4001 Roberts Point Rd., Sarasota, FL 33581, 349-0200.**

## Aurora, IL

Microcomputer systems for home or business; peripherals, software, books & magazines. Apple, Hewlett-Packard, North Star, Cromemco systems. IDS-440G printer w/Apple graphics, New HP-85 & HP calculators. **Farnsworth Computer Center, 1891 N. Farnsworth Ave., Aurora, IL 60505, 851-3888.**

## Chicago, IL

Brand new lowest prices, never undersold, postpaid in USA—Teletype 43 keyboard printers, Okidata & Integral Data printers, SS-50 bus computers, peripherals & business software. **Data Mart, 914 East Waverly Street, Arlington Heights, IL 60004, 398-8525.**

## Garden City, MI

Complete systems for business, professional and personal applications. Custom programming available. Apple II, North Star, Vector Graphic and other lines of microcomputers, software, books, components. **Computer Center, 28251 Ford Rd., Garden City, MI 48135, 422-2570.**

**Dealers:** Listings are **$15** per month in prepaid quarterly payments, or one yearly payment of **$150**, also prepaid. Ads include 25 words describing your products and services plus your company name, address and phone. (No area codes or merchandise prices, please.) Call Marcia at 603-924-7138 or write *Kilobaud Microcomputing*, Ad Department, Peterborough, NH 03458.

## Westland, MI

Integrated circuits, TTL, CMOS, linear. Many hard to find "S" and "LS" types. Resistors, capacitors, diodes, IC sockets and many other items. **Westland Electronics, 34245 Ford Rd., Westland, MI 48185, 728-0650.**

## Hannibal, MO

Ohio Scientific products, modifications, service, software. 8" disk for C1p, C4p. Process control specialist. **E&I Technical Service, 5300 Paris Gravel Road, Hannibal, MO 63401, 248-0084.**

## St. Louis, MO

Experimenters' Paradise. Electronic and mechanical components. Computer People, Audio People, Hams, Robot Builders, Experimenters. Open six days a week. **Gateway Electronics Corp., 8123-25 Page Blvd., St. Louis, MO 63130, 427-6116.**

## Portland, OR

Ohio Scientific specialists for business and personal computers. Local service. Terminals, printers, custom programming. Full OSI product line on display! 10 AM to 6 PM M-F. **Fial Computer, 11266 SE 21st Ave., Milwaukie, OR 97222, 654-9574.**

## Kingston, PA

We support Level II and Model II. Books, magazines, programs, parts, accessories, peripherals, free literature, free seminars, cassettes, floppies, filters, transformers, caps, chips, CRTs. **Artco Electronics, 302 Wyoming Ave., Kingston, PA 18704, 287-1014.**

## Sara, Mexico

Learn how to utilize and program the Z-80 microprocessor to maintain your company's records in top shape. This technique has just been introduced to Mexico. Courses, applications, maintenance, service. **Digitales, S.A. de C.V. Sara 4612, Mexico 14, D.F., 5-17-41-59.**

# MICRO QUIZ

*from page 19*

Answer: 3

J = 1 => X(7) = 1
J = 2 => X(1) = 2
J = 3 => X(7) = 3
J = 4 => X(2) = 4
J = 5 => X(2) = 5
J = 6 => X(1) = 6

(The next quiz question tests your knowledge of digital electronics.)

# CLASSIFIEDS

Classified advertisements are intended for use by persons desiring to buy, sell or trade used computer equipment. No commercial ads are accepted.

Two sizes of ads are available. The $5 box allows up to 5 lines of about 35 characters per line, including spaces and punctuation. The $10 box allows up to 10 lines. Minimize use of capital letters to save space. No special layouts allowed. Payment is required in advance with ad copy. We cannot bill or accept credit.

Advertising text and payment must reach us 60 days in advance of publication (i.e., copy for March issue, mailed in February, must be here by Jan. 1). The publisher reserves the right to refuse questionable or inapplicable advertisements. Mail copy with payment to: **Classifieds, Kilobaud Microcomputing, Peterborough NH 03458.** Do not include any other material with your ad as it may be delayed.

For Sale: One Gimix computer nearly new, excellent condition. 6800 CPU card, 16K static memory card (Gimix). $800. Paul Lamar, 123 S. Juanita St., Redondo Beach, CA 90277. Work 213-374-1673, home 213-316-8351.

For Sale: Smoke Signal Broadcasting disk controller card, with the following software: on 8" floppy disk, text editing system, disk file BASIC and mnemonic assembler. $250. Paul Lamar, 123 S. Juanita St., Redondo Beach, CA 90277. Work 213-374-1673, home 213-316-8351.

Trade ham station for AIM 65. Also trading O.S.I. programs, etc. Write to Henry A. Etchason, Box 147, Sage, AR 72573.

TRS-80 Mod. I, Level I with Expansion Interface & 10 key, $1000. IBM Selectric 71-3 I/O with TRS-80 print port interface, $650. Will sell both for $1400. D. Bowie, Box 3453, San Francisco, CA 94119, 415-861-6883.

For Sale: S. D. Expandoram, S-100, 16K with 4115s, assembled, never used. $175 or best offer. Call 701-968-4525, or write T. Cartwright, Box 301, Cando, ND 58324.

For Sale: Back issues of Byte, all my beginning computer books and cassette learning tapes, assorted electronic books, etc. Send SASE for list to Don Hastings, Box 366, Hemingway, SC 29554.

For Sale: Heath H8 w/16K, H8-5. H9 w/ GRAFIX and page erase. Heath cassette recorder, software, complete documentation. Complete system, up and running, $650. Dave Leupp, 852 Westover Terr., Whiteman AFB, MO 65305, 816-563-2208.

For Sale: TRS-80, Level II, 16K w/ many extras, $600. Mint condition. Kilobaud issues #1 to present. Best offer. Larry Guerrera, Box 933, Woodside, NY 11377, 212-545-4917.

For Sale: Eaton LRC 7000+ 64 char. printer, $250. Radio Shack Quick Printer II, 32 char. (Cat. No. 26-1155), $150. Send certified check or money order: Wm. R. Spencer, Jr., 5421 Grandin Rd. Ext., Salem, VA 24153.

Wanted: 610 expansion board for OSI C1P. With or without RAM chips. Send price and description to Gary Rinehart, RR#3, North Manchester, IN 46962.

For Sale—SWTP MP-16, 16K memory board, excellent condition. Paid over $450. Will sell for $299 or make offer. Call 702-453-4775 after 5 pm.

For Sale: SWTP 6800 w/ 32K, SSB dual disk drive, CT-64 & VM-64, Percom cassette interface, manual and software and extra serial and parallel cards. $3350. Will separate. John Rogers, 312-549-0008 after 5 pm.

For Sale: Unused Polymorphic System 8813 with 32K RAM memory with floppy-disk drives, printer interface and Abern-Sopher Multiwriter III. System has Canadian import tax paid. Offers for complete system to: Bishop Management, #8—825 McBride Blvd., New Westminster, B.C., Canada, V3L 5B5. 604-525-8148.

For Sale: MSI-6800 microcomputer with 16K + RAM, 8K PROM board, two serial ports, two cassette ports (J.P.C.), new Percom 5" disk with controller card, lots of disk software. This system has been used very little, $1700. KIM-1 with enclosure and all manuals, $110. Meca-Beta high-speed cassette recorder, 9600 baud, RS-232, $325. Micon 32 char. terminal, 300 baud, RS-232, $325. Everything for $2300 and you ship. Bill Pinkerton, 129–2nd Ave., Indialantic, FL 32903. 305-725-7016.

PET 2nd cassette ($60) + more PET equip. Lots of software + manuals. For sale (cheap!) or trade. Send SASE to D. Coles, 3713 Bay To Bay, Tampa, FL 33609.

Wanted: OSI b&w video board #540. If you upgraded to color you have one gathering dust. Mike Schroeder, 1251 Minnesota Ave., Detroit Lakes, MN 56501.

Sell: Heath system complete and assembled H8, H9, 40K memory, H17 disk, 2 drives, DEC writer, 2 cassette recorders. All programs documentation and disks. Only $3500. Every item below kit price. E. McCormick, 8 Monmouth Shire Lane, Spring Lake, NJ 07762. 201-449-4093 after 7:00 pm.

For Sale: Xitan Z-80 system. Mainframe, ZPU, SMB, VDB, 48K RAM, 16K ROM (12K BASIC in ROM), keyboard, manuals, software. Complete system $1800. Terry Young, 4 Aiken St., Derry, NH 03038, 603-434-0257.

Wanted: Synertek KTM-2 keyboard and/or Trendcom 200 printer. In return I will fabricate an equal value of prototype PC boards. R. Hegel, 7332 Portland Ave., Richfield, MN 55423.

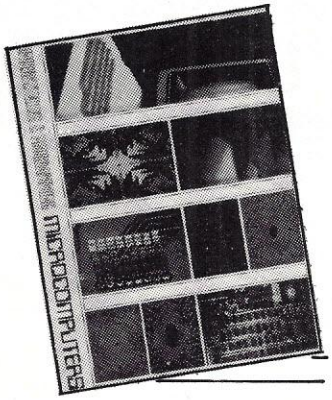
TRS-80 Quick Printer-II plus nine rolls paper and cables for keyboard and expansion interface, $180. Like new, list over $240. G. Atkinson, Box 40387, San Francisco, CA 94140, 415-647-9122.

Diablo Hytype 1 Model 1200. Best of the "daisy wheel" printers. Brand new units w/pin feed friction platen & print wheel. Interface for Apple, TRS-80 & CP/M systems, maintenance manual and additional interface info available. There is no better buy anywhere. After 6 PM. Scott Priester, 211 White Water Ct., Greer, SC 29651, 803-268-0678.

✓ Reader Service index—page 241

# kb microcomputing book nook

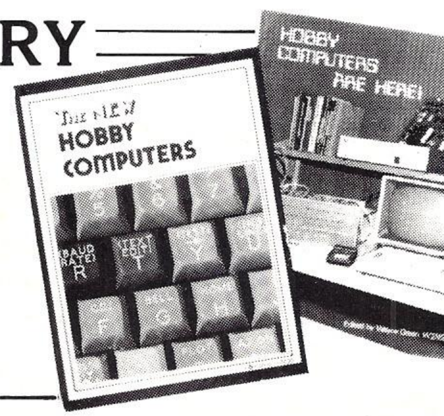
# 2 NEW BOOKS FROM KB MICROCOMPUTING


● **40 COMPUTER GAMES**—BK7381—Forty games in all in nine different categories. Games for large and small systems, and even a section on calculator games. Many versions of BASIC used and a wide variety of systems represented. A must for the serious computer gamesman. $7.95*

● **UNDERSTANDING AND PROGRAMMING MICROCOMPUTERS** —BK7382—A valuable addition to your computing library. This two part text includes the best articles that have appeared in 73 and Kilobaud Microcomputing magazines on the hardware and software aspects of the new microcomputing hobby. Well known authors and well structured text helps the reader get involved in America's fastest growing hobby. $10.95*

## INTRODUCTORY




● **HOBBY COMPUTERS ARE HERE!**—BK7322—If you (or a friend) want to come up to speed on how computers work . . . hardware and software . . . this is an excellent book. It starts with the fundamentals and explains the circuits, and the basics of programming. This book has the highest recommendations as a teaching aid for newcomers. $4.95.*

● **THE NEW HOBBY COMPUTERS**—BK7340—This book takes it from where "HOBBY COMPUTERS ARE HERE!" leaves off, with chapters on Large Scale Integration, how to choose a microprocessor chip, an introduction to programming, low cost I/O for a computer, computer arithmetic, checking memory boards . . . and much, much more! Don't miss this tremendous value! Only $4.95.*

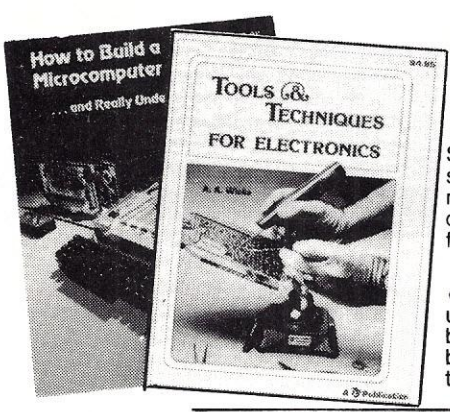
## *INTRODUCTION TO MICROCOMPUTERS (VOL. 0→III)*

● **AN INTRODUCTION TO MICROCOMPUTERS, VOL. 0**—BK1130—The Beginner's Book—Written for readers who know nothing about computers—for those who have an interest in how to use computers—and for everyone else who must live with computers and should know a little about them. The first in a series of 4 volumes, this book will explain how computers work and what they can do. Computers have become an integral part of life and society. During any given day you are affected by computers, so start learning more about them with Volume 0. $7.95.*

● **VOL. I**—BK1030—**2nd Edition completely revised.** Dedicated to the basic concepts of microcomputers and hardware theory. The purpose of Volume I is to give you a thorough understanding of what microcomputers are. From basic concepts (which are covered in detail), Volume I builds the necessary components of a microcomputer system. This book highlights the difference between minicomputers and microcomputers. $12.50.*

● **VOL. II**—BK1040 (with binder)—$30.00* —Contains descriptions of individual microprocessors and support devices used only with the parent microprocessor. Volume II describes all available chips.

● **VOL. III**—BK1133 (with binder)—$20.00.* Contains descriptions of all support devices that can be used with any microprocessor.




● **HOW TO BUILD A MICROCOMPUTER—AND REALLY UNDERSTAND IT**—BK7325—by Sam Creason. The electronics hobbyist who wants to build his own microcomputer system now has a practical "How-To" guidebook. This book is a combination technical manual and programming guide that takes the hobbyist step-by-step through the design, construction, testing and debugging of a complete microcomputer system. Must reading for anyone desiring a true understanding of small computer systems. $9.95.*

● **TOOLS & TECHNIQUES FOR ELECTRONICS**—BK7348—by A. A. Wicks is an easy-to-understand book written for the beginning kit builder as well as the experienced hobbyist. It has numerous pictures and descriptions of the safe and correct ways to use basic and specialized tools for electronic projects as well as specialized metal working tools and the chemical aids which are used in repair shops. $4.95.*

*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to Kilobaud *Microcomputing* Book Department • Peterborough NH 03458. Be sure to include check or detailed credit card information. No C.O.D. orders accepted. All orders add $1.00 handling. Please allow 4-6 weeks for delivery. Questions regarding your order? Please write to Customer Service at the above address.

**FOR TOLL FREE ORDERING CALL 1-800-258-5473**

# kb microcomputing book nook

## SPECIAL INTERESTS


● **TRS-80 DISK AND OTHER MYSTERIES** – BK1181 – by Harvard C. Pennington. This is the definitive work on the TRS-80 disk system. It is full of detailed "How to" information with examples, samples and in-depth explanations suitable for beginners and professionals alike. The recovery of one lost file is worth the price alone. $22.50.*

● **INTRODUCTION TO TRS-80 GRAPHICS** – BK1180 – by Don Inman. Dissatisfied with your Level I or Level II manual's coverage of graphics capabilities? This well-structured book (suitable for classroom use) is ideal for those who want to use all the graphics capabilities built into the TRS-80. A tutorial method is used with many demonstrations. It is based on the Level I, but all material is suitable for Level II use. $8.95.*

● **MICROPROCESSOR INTERFACING TECHNIQUES** – BK1037 – by Austin Lesea & Rodnay Zaks will teach you how to interconnect a complete system and interface it to all the usual peripherals. It covers hardware and software skills and techniques, including the use and design of model buses such as the IEEE 488 or S100. $15.95.*

● **MICROPROCESSOR LEXICON – ACRONYMS AND DEFINITIONS** – BK1137 – Compiled by the staff of SYBEX is a convenient reference in pocket size format. Sections include acronyms and definitions, part numbers and their definitions, S-100 signals, RS232 signals, IEEE 499 signals, microcomputer and microprocessors, JETDS summary (military) and a code conversion table. $2.95.*

● **MICROPROCESSORS FROM CHIPS TO SYSTEMS** – BK1036 – by Rodnay Zaks is a complete and detailed introduction to microprocessors and microcomputer systems. No preliminary knowledge of computers or microprocessors is required to read this book, although a basic engineering knowledge is naturally an advantage. Intended for all wishing to understand the concepts, techniques and components of microprocessors in a short time. $10.95.*

## MONEYMAKING


● **HOW TO MAKE MONEY WITH COMPUTERS** – BK1003 – In 10 information-packed chapters, Jerry Felsen describes more than 30 computer-related, money-making, high profit, low capital investment opportunities. $15.00.*

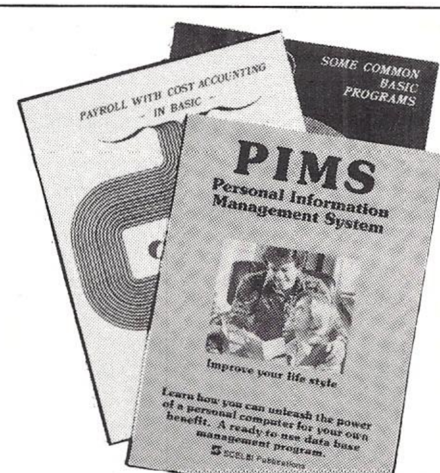
● **HOW TO SELL ANYTHING TO ANYBODY** – BK7306 – According to *The Guinness Book of World Records*, the author, Joe Girard, is "the world's greatest salesman." This book reveals how he made a fortune – and how you can, too. $2.25.*

● **FREELANCE SOFTWARE PUBLISHING** – BK1179 – by B. J. Korites. "This book is about money and how to make it by writing and selling computer programs," (author's foreword). If you have the skills to write a saleable program, you now need to acquire the skills to sell that program. This compact book comprehensively covers the entire publishing process and many aspects of software salesmanship. $14.95.*

● **THE INCREDIBLE SECRET MONEY MACHINE** – BK1178 – by Don Lancaster. A different kind of "cookbook" from Don Lancaster. Want to slash taxes? Get free vacations? Win at investments? Make money from something that you *like* to do? You'll find this book essential to give you the key insider details of what is really involved in starting up your own money machine. $5.95.*

## BUSINESS

● **PAYROLL WITH COST ACCOUNTING – IN BASIC** – BK1001 – by L. Poole & M. Borchers, includes program listings with remarks, descriptions, discussions of the principle behind each program, file layouts, and a complete user's manual with step-by-step instructions, flowcharts, and simple reports and CRT displays. Payroll and cost accounting features include separate payrolls for up to 10 companies, time-tested interactive data entry, easy correction of data entry errors, job costing (labor of distribution), check printing with full deduction and pay detail, and 16 different printed reports, including W-2 and 941 (in **CBASIC**). $20.00.*




● **SOME COMMON BASIC PROGRAMS** – BK1053 – published by Adam Osborne & Associates, Inc. Perfect for non-technical computerists requiring ready-to-use programs. Business programs, plus miscellaneous programs. Invaluable for the user who is not an experienced programmer. All will operate in the stand-alone mode. $12.50 paperback.*

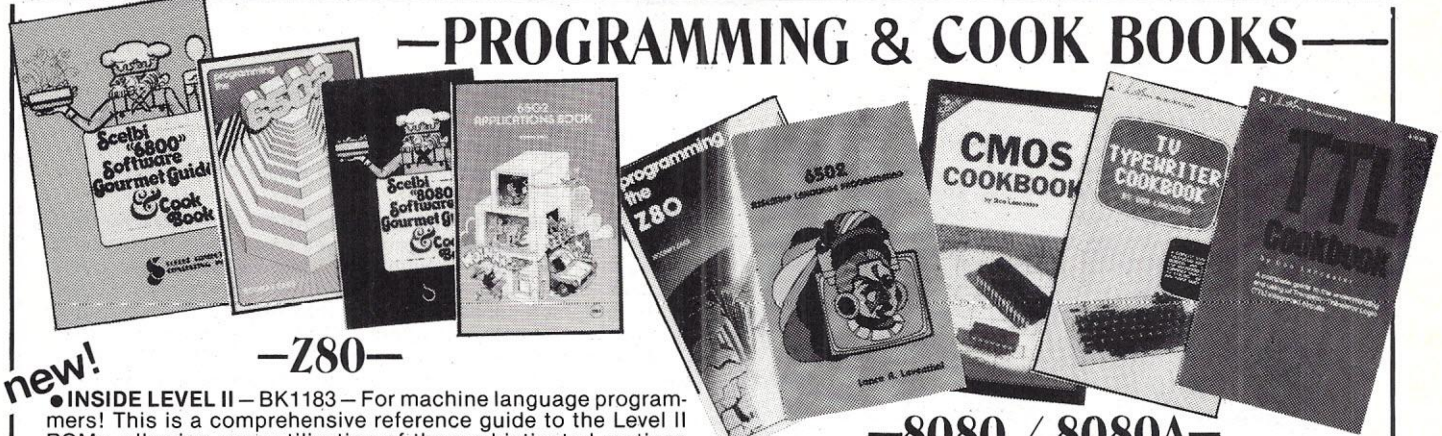
● **PIMS: PERSONAL INFORMATION MANAGEMENT SYSTEM** – BK1009 – Learn how to unleash the power of a personal computer for your own benefit in this ready-to-use data-base management program. $11.95.*

*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to Kilobaud *Microcomputing* Book Department • Peterborough NH 03458. Be sure to include check or detailed credit card information. No C.O.D. orders accepted. All orders add $1.00 handling. Please allow 4-6 weeks for delivery. Questions regarding your order? Please write to Customer Service at the above address.

**FOR TOLL FREE ORDERING CALL 1-800-258-5473**

# kb microcomputing book nook

## —PROGRAMMING & COOK BOOKS—




## —Z80—

new!

● **INSIDE LEVEL II** — BK1183 — For machine language programmers! This is a comprehensive reference guide to the Level II ROMs, allowing easy utilization of the sophisticated routines they contain. It concisely explains set-ups, calling sequences, variable passage and I/O routines. Part II presents an entirely new composite program structure which unloads under the SYSTEM command and executes in both BASIC and machine code with the speed and efficiency of a compiler. Special consideration is given to disk systems. $15.95.*

● **PROGRAMMING THE Z-80** — BK1122 — by Rodnay Zaks. Here is assembly language programming for the Z-80 presented as a progressive, step-by-step course. This book is both an educational text *and* a self-contained reference book, useful to both the beginning and the experienced programmer who wish to learn about the Z-80. Exercises to test the reader are included. $14.95.*

● **Z-80 ASSEMBLY LANGUAGE PROGRAMMING** — BK1177 — by Lance A. Leventhal. This book thoroughly covers the Z80 instruction set, abounding in simple programming examples which illustrate software development concepts and actual assembly language usage. Features include Z80 I/O devices and interfacing methods, assembler conventions, and comparisons with 8080A/8085 instruction sets and interrupt structure. $12.50.*

● **Z-80 SOFTWARE GOURMET GUIDE AND COOKBOOK** — BK1045 — by Nat Wadsworth. Scelbi's newest cookbook! This book contains a complete description of the powerful Z-80 instruction set and a wide variety of programming information. Use the author's ingredients including routines, subroutines and short programs, choose a time-tested recipe and start cooking! $15.95.*

## —6502—

● **PROGRAMMING THE 6502 (Second Edition)** — BK1005 — Rodnay Zaks has designed a self-contained text to learn programming, using the 6502. It can be used by a person who has never programmed before, and should be of value to anyone using the 6502. The many exercises will allow you to test yourself and practice the concepts presented. $12.95.*

● **6502 APPLICATIONS BOOK** — BK1006 — Rodnay Zaks presents practical-application techniques for the 6502 microprocessor, assuming an elementary knowledge of microprocessor programming. You will build and design your own domestic-use systems and peripherals. Self-test exercises included. $12.95.*

● **6502 ASSEMBLY LANGUAGE PROGRAMMING** — BK1176 — by Lance A. Leventhal. This book provides comprehensive coverage of the 6502 microprocessor assembly language. Leventhal covers over 80 programming examples from simple memory load loops to complete design projects. Features include 6502 assembler conventions, input/output devices and interfacing methods, and programming the 6502 interrupt system. $12.50.*

● **6502 SOFTWARE GOURMET GUIDE AND COOKBOOK** — BK1055 — by Robert Findley. This book introduces the BASIC language programmer into the realm of machine-language programming. The description of the 6502 structure and instruction set, various routines, subroutines and programs are the ingredients in this cookbook. "Recipes" are included to help you put together exactly the programs to suit your taste. $12.95.*

## —8080 / 8080A—

● **8080A/8085 Assembly Language Programming** — by Lance Leventhal — BK1004 — Assembly language programming for the 8080A/8085 is explained with a description of the functions of assemblers and assembly instructions, and a discussion of basic software development concepts. Many fully debugged, practical programs are included as is a special section on structured programming. $12.50.*

● **8080 PROGRAMMING FOR LOGIC DESIGN** — BK1078 — Ideal reference for an indepth understanding of the 8080 processor. Application-oriented and the 8080 is discussed in light of replacing conventional, hard-wired logic. Practical design considerations are provided for the implementation of an 8080-base control system. $9.50.*

● **8080 SOFTWARE GOURMET GUIDE AND COOKBOOK** — BK1102 — If yu have been spending too much time developing simple routines for your 8080, try this new book by Scelbi Computing and Robert Findley. Describes sorting, searching, and many other routines for the 8080 user. $12.95.*

## —6800—

● **6800 PROGRAMMING FOR LOGIC DESIGN** — BK1077 — Oriented toward the industrial user, this book describes the process by which conventional logic can be replaced by a 6800 microprocessor. Provides practical information that allows an experimenter to design a complete micro control system from the "ground up." $9.50.*

● **6800 SOFTWARE GOURMET GUIDE AND COOKBOOK** — BK1075 — Like its culinary cousin, *The 8080 Gourmet Guide*, this book by Scelbi Computing and Robert Findley describes sorting, searching and other routines — this time for the 6800 user. $12.95.*

## —COOK BOOKS—

● **CMOS COOKBOOK** — BK1011 — by Don Lancaster. Details the application of CMOS, the low power logic family suitable for most applications presently dominated by TTL. Required reading for every serious digital experimenter! $10.50.*

● **TVT COOKBOOK** — BK1064 — by Don Lancaster. Describes the use of a standard television receiver as a microprocessor CRT terminal. Explains and describes character generation, cursor control and interface information in typical, easy-to-understand Lancaster style. $9.95.*

● **TTL COOKBOOK** — BK1063 — by Donald Lancaster. Explains what TTL is, how it works, and how to use it. Discusses practical applications, such as a digital counter and display system, events counter, electronic stopwatch, digital voltmeter and a digital tachometer. $9.50.*

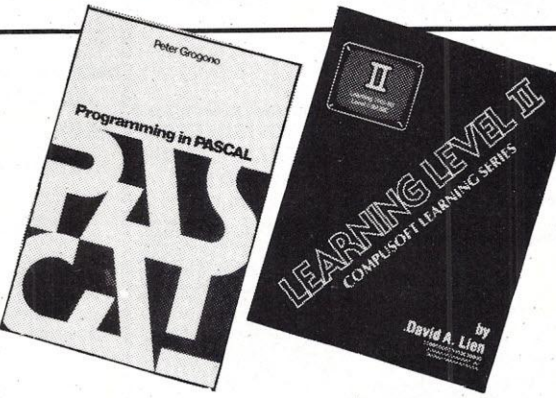
● ***MICROCOMPUTING* CODING SHEETS** *Microcomputing*'s dozen or so programmers wouldn't try to work without these handy scratch pads, which help prevent the little errors that can cost hours and hours of programming time. Available for programming is Assembly/Machine Language (PD1001), which has columns for address, instruction (3 bytes), source code (label, op code, operand) and comments; and for BASIC (PD1002) which is 72 columns wide. 50 sheets to a pad. $2.39.*

*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to Kilobaud *Microcomputing* Book Department • Peterborough NH 03458. Be sure to include check or detailed credit card information. No C.O.D. orders accepted. All orders add $1.00 handling. Please allow 4-6 weeks for delivery. Questions regarding your order? Please write to Customer Service at the above address.

**FOR TOLL FREE ORDERING CALL 1-800-258-5473**

# kb microcomputing book nook

## BASIC AND PASCAL



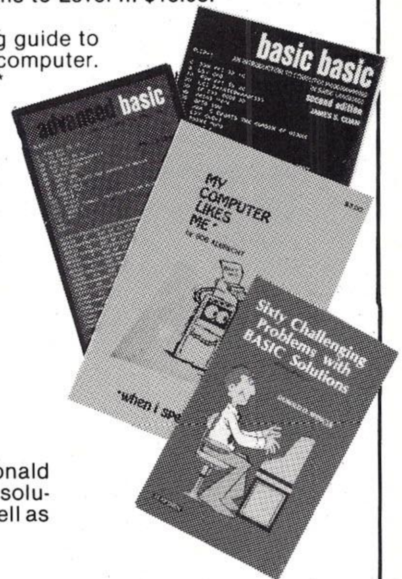

***NEW REVISED EDITION***

● **PROGRAMMING IN PASCAL**—BK1140—by Peter Grogono. The computer programming language PASCAL was the first language to embody in a coherent way the concepts of structured programming, which has been defined by Edsger Dijkstra and C.A.R. Hoare. As such, it is a landmark in the development of programming languages. PASCAL was developed by Niklaus Wirth in Zurich; it is derived from the language ALGOL 60 but is more powerful and easier to use. PASCAL is now widely accepted as a useful language that can be efficiently implemented, and as an excellent teaching tool. It does not assume knowledge of any other programming language; it is therefore suitable for an introductory course. $12.95.*

● **THE BASIC HANDBOOK**—BK1174—by David Lien. This book is unique. It is a virtual ENCYCLOPEDIA of BASIC. While not favoring one computer over another, it explains over 250 BASIC words, how to use them and alternate strategies. If a computer does not possess the capabilities of a needed or specified word, there are often ways to accomplish the same function by using another word or combination of words. That's where the HANDBOOK comes in. It helps you get the most from your computer, be it a "bottom-of-the-line" micro or an oversized monster. $14.95.*

● **LEARNING LEVEL II**—BK1175—by David Lien. Written especially for the TRS-80, this book concentrates on Level II BASIC, exploring every important BASIC language capability. Updates are included for those who have studied the Level I User's Manual. Sections include: how to use the Editor, dual cassette operation, printers and peripheral devices, and the conversion of Level I programs to Level II. $15.95.*


● **BASIC NEW 2ND EDITION**—BK1081—by Bob Albrecht. Self-teaching guide to the computer language you will need to know for use with your microcomputer. This is one of the easiest ways to learn computer programming. $6.95.*



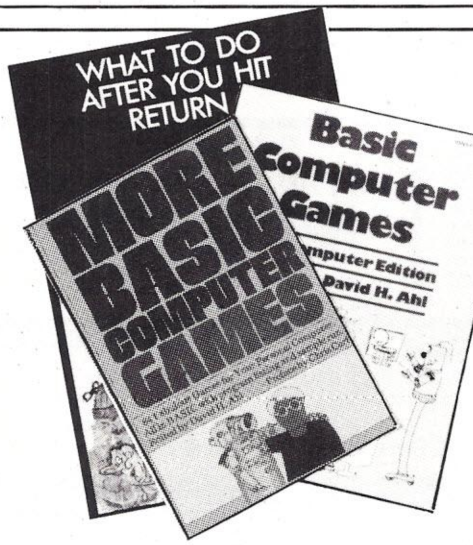

● **BASIC BASIC (2ND EDITION)**—BK1026—by James S. Coan. This is a textbook which incorporates the learning of computer programming using the BASIC language with the teaching of mathematics. Over 100 sample programs illustrate the techniques of the BASIC language and every section is followed by practical problems. This second edition covers character string handling and the use of data files. $9.45.*

● **ADVANCED BASIC**—BK1000—Applications, including strings and files, coordinate geometry, area, sequences and series, simulation, graphing and games. $9.65*.

● **MY COMPUTER LIKES ME . . . WHEN I SPEAK BASIC**—BK1039—An introduction to BASIC . . . simple enough for kids. If you want to teach BASIC to anyone quickly, this is the way to go. $3.95.*

● **SIXTY CHALLENGING PROBLEMS WITH BASIC SOLUTIONS (2nd Edition)**—BK1073—by Donald Spencer, provides the serious student of BASIC programming with interesting problems and solutions. No knowledge of math above algebra required. Includes a number of game programs, as well as programs for financial interest, conversions and numeric manipulations. $6.95.*

## GAMES




● **WHAT TO DO AFTER YOU HIT RETURN**—BK1071—PCC's first book of computer games . . . 48 different computer games you can play in BASIC . . . programs, descriptions, many illustrations. Lunar Landing, Hammurabi, King, Civel 2, Qubic 5, Taxman, Star Trek, Crash, Market, etc. $10.95.*

● **BASIC COMPUTER GAMES**—BK1074—Okay, so once you get your computer and are running in BASIC, then what? Then you need some programs in BASIC, that's what. This book has 101 games for you from very simple to real buggers. You get the games, a description of the games, the listing to put in your computer and a sample run to show you how they work. Fun. Any one game will be worth more than the price of the book for the fun you and your family will have with it. $7.50.*

new!

● **MORE BASIC COMPUTER GAMES**—BK1182—edited by David H. Ahl. More fun in BASIC! 84 new games from the people who brought you *BASIC Computer Games.* Includes such favorites as Minotaur (battle the mythical beast) and Eliza (unload your troubles on the doctor at bargain rates). Complete with game description, listing and sample run. $7.50.*

*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to Kilobaud *Microcomputing* Book Department • Peterborough NH 03458. Be sure to include check or detailed credit card information. No C.O.D. orders accepted. All orders add $1.00 handling. Please allow 4–6 weeks for delivery. Questions regarding your order? Please write to Customer Service at the above address.

**FOR TOLL FREE ORDERING CALL 1-800-258-5473**

HEX ENCODED KEYBOARD
Four onboard LEDs indicate the HEX code generated for each key depression. The board requires a single +5 volt supply. Board only $15.00 Part No. HEX-3, with parts $49.95 Part No. HEX- 3A. 44 pin edge connector $4.00 Part No. 44P.
ASCII TO CORRESPONDENCE CODE CONVERTER
This bidirectional board is a direct replacement for the board inside the Trendata 1000 terminal. The on board connector provides RS-232 serial in and out. Sold only as an assembled and tested unit for $249.95. Part No. TA 1000C
T.V. INTERFACE
● Converts video to AM modulated RF, Channels 2 or 3. So powerful almost no tuning is required. On board regulated power supply makes this extremely stable. Rated very highly in Doctor Dobbs' Journal. Recommended by Apple ● Power required is 12 volts AC C.T., or +5 volts DC ● Board only $7.60 part No. 107, with parts $13.50 Part No. 107A
TAPE INTERFACE
● Converts a low cost tape recorder to a digital recorder ● Works up to 1200 baud ● Digital in and out are TTL-serial ● Output of board connects to mic. in of recorder ● Earphone of recorder connects to input on board ● No coils ● Requires +5 volts, low power drain ● Board only $7.60 Part No. 111, with parts $29.95 Part No. 111A
ASCII KEYBOARD
53 Keys popular ASR-33 format • Rugged G-10 P. C. Board • Tri-mode MOS encoding • Two-Key Rollover • MOS/DTL/TTL Compatible • Upper Case lockout • Data and Strobe inversion option • Three User Definable Keys • Low contact bounce • Selectable Parity • Custom Keycaps • George Risk Model 753. Requires +5, -12 volts. $59.95 Kit.
T.V. TYPEWRITER
● Stand alone TVT ● 32 char/line, 16 lines, modifications for 64 char/line included ● Parallel ASCII (TTL) input ● Video output ● 1K on board memory ● Output for computer controlled curser ● Auto scroll ● Non-destructive curser ● Curser inputs: up, down, left, right, home, EOL, EOS ● Scroll up, down ● Requires +5 volts at 1.5 amps, and -12 volts at 30 mA ● All 7400, TTL chips ● Char. gen. 2513 ● Upper case only ● Board only $39.00 Part No. 106, with parts $145.00 Part No. 106A
UART & BAUD RATE GENERATOR
● Converts serial to parallel and parallel to serial ● Low cost on board baud rate generator ● Baud rates: 110, 150, 300, 600, 1200, and 2400 ● Low power drain +5 volts and -12 volts required ● TTL compatible ● All characters contain a start bit, 5 to 8 data bits, 1 or 2 stop bits, and either odd or even parity. ● All connections go to a 44 pin gold plated edge connector ● Board only $12.00 Part No. 101, with parts $35.00 Part No. 101A, 44 pin edge connector $4.00 Part No. 44P
ASCII KEYBOARD
TTL & DTL compatible • Full 67 key array • Full 128 character ASCII output • Positive logic with outputs resting low • Data Strobe • Five user-definable spare keys • Standard 22 pin dual card edge connector • Requires +5VDC, 325 mA. Assembled & Tested. Cherry Pro Part No. P70-05AB. $119.95.
SOROC IQ 120
Upper/lower case display • Numeric keypad & cursor keys • Protected fields, ½ intensity display • RS 232 interface & aux. port. IQ120—$799.95 • IQ140 Detachable keyboard—$1199.95
MODEM
● Type 103 ● Full or half duplex ● Works up to 300 baud ● Originate or Answer ● Serial TTL input and output ● connect 8 Ω speaker and crystal mic. directly to board ● Requires +5 volts ● Board only $7.60 Part No. 109, with parts $29.95 Part No. 109A.
A-to-D D-to-A CONVERTER
Analog to Digital, Digital to Analog Converter, A-D conversion time 20us. D-A conversion 5us. Uses include speech and music synthesizing and slow scan TV. Single power supply (5V), 8 Bits wide, latched I/O, strobe lines. Part No. 79287K Complete Kit $49.95 • Part No. 79287A Assembled $69.95
44 BUS MOTHER BOARD
Has provisions for ten 44 pin (.156) connectors, spaced 3/4 of an inch apart. Pin 20 is connected to X, and 22 is connected to Z for power and ground. All the other pins are connected in parallel. This board also has provisions for bypass capacitors. Board cost $15.00 Part No. 102. Connectors $3.00 each Part No. 44WP.
RS-232/20mA INTERFACE
This board has two passive, opto-isolated circuits. One converts RS-232 to 20mA, the other converts 20mA to RS-232. All connections go to a 10 pin edge connector. Requires +12 and -12 volts. Board only $9.95, part no. 7901, with parts $14.95 Part No. 7901A.
RS-32/TTL INTERFACE
● Converts TTL to RS-232, and converts RS-232 to TTL ● Two separate circuits ● Requires -12 and +12 volts ● All connections go to a 10 pin edge connector, kit $9.95 Part No. 232A 10 Pin edge connector $3.00 part No. 10P.
COMPUCOLOR II
With reg. keyboard MOD3 8K $1449.95 MOD4 16 K $1495.95 MOD5 32K $1699.95 Without disk drive subtract $450.00. Add-on drives, $495.00. With 101 key option add $134.95. With 117 key option add $179.95.
SOLID STATE SWITCH
Your computer can control power (120VAC) to your printer, lights, and other 120VAC appliances up to 720 watts (6AMPS at 120VAC). Input 3 to 15 VDC, 2-13 MA TTL compatible, isolation 1500V. Part No. 79000K 1 Channel Kit $9.95 • Assm. $12.50 • Part No. 79004K 4 Channel Kit $34.95 • Assm. $44.95.
SUPER MODEM
Orignate, RS-232 and 20 mA compatable, Full duplex, and half duplex, direct connect or acoustic coupled, on board power supply, carrier detect light, DB25 plug, 300 BAUD, Type 103 compatable frequencies, Bare board Part No. 2000, $19,95, Kit Part No. 2000A, $99.95.
DC POWER SUPPLY
• Board supplies a regulated +5 volts at 3 amps., +12, -12, and -5 volts at 1 amp. • Power required is 8 volts AC at 3 amps., and 24 volts AC C.T. at 1.5 amps. • Board only $12.50 Part No. 6085, with parts excluding transformers $42.50 Part No. 6085A
Send for FREE Catalog...a big self addressed envelope with 80¢ postage gets it fastest!
To Order: Mention part no., description, and price. In USA shipping paid by us for orders accompanied by check or money order. We accept C.O.D. orders (U.S. only) or a VISA or Master Charge no., expiration date, signature and phone no., shipping charges will be added. CA residents add 6.5% for tax. Outside USA add 15% for air mail postage and handling. Payment must be in U.S. dollars. Dealer inquiries invited. Prices subject to change without notice.
VISA
master charge
Order Line: (408) 448-0800
47
ELECTRONIC SYSTEMS Dept. KB, P.O. Box 21638, San Jose, CA USA 95151

New Summer 1980
SEND FOR OUR FREE CATALOG

# CompuMart lets you put 'em on trial

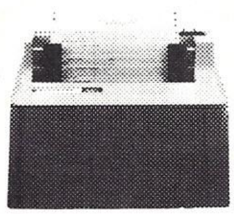
CompuMart has been selling computers by mail since 1971. Our thousands of satisfied customers rely on CompuMart for services not generally available from the others. Namely:

- Product Selection/Each product advertised by CompuMart has been evaluated by our in-house staff for best price, performance, and supplier reliability
- Return Privilege/After receipt of our products, you are protected by CompuMart's exclusive, 10-day return priviledge- good for all products except software.
- Support/Our Customer Service Dept. and expert technicians are always there to assist you by phone or at CompuMart's outlets. Our knowledgeable phone sales force can provide you with detailed information and complete product specifications.
- Phone Ordering/For added convenience, CompuMart maintains a toll-free ordering number. 1-800-343-5504.

# with 10 day free return

## Printers

### The Paper Tiger Printer From Integral Data



Uses standard ⅜ inch roll paper and ribbon
40 characters per line
Speed: 40 characters per second
UL approved
High resolution dot matrix impact printer

**IDS Paper Tiger Printer** .................... **$995**
**IDS Graphics Paper Tiger Printer** ........... **$1,094**

### NEW! From Integral Data. The IDS 460.

We saw this new desktop printer at the NCC 80 and when we saw its features: Correspondence quality printing, High-resolution graphics capability, programmable print control functions, and automatic text justification—we knew that we had to offer this printer to our cost/features conscious customers $1,295

### CENTRONICS PRINTERS

New! The incredible Model 737- Correspondence and Draft Quality Printing for Under $1,000. This is the first printer in its class to offer print quality suitable for text processing, plus the performance and application flexibility required for data processing. **$995**

737-3 (Serial Interface) **$1,045**

Tractor Feed Printer- Centronics' Most Popular Model. Perfect for the needs of a small business sytesm. Recommended by Apple and Radio Shack. **$1,079**

### NEC The First Name in Letter Quality Printers.

CompuMart offers beautiful print quality with NEC Spinwriter terminals. The Spinwriters, both KSR and RO versions, give unsurpassed hard copy output. CompuMart offers a complete range of NEC Spinwriters—Call our expert salesforce

### The Omni 810 Printer from Texas Instruments

TI Omni 820 Receive-Only (RO) Package. Includes machine-mounted paper tray and cable. A compressed print option and device forms control are standard features **$2,155**

TI Omni 820 Keyboard Send Receive (KSR) Package Comes with full ASCII Keyboard with numeric Keypad and an EIA cable with autospeed select. **$2,395**

## Monitors

### EXCLUSIVE from CompuMart!

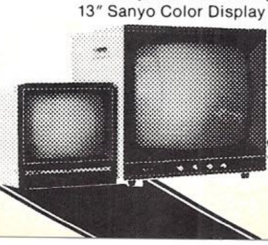
Special Offer. Zenith Color Video Monitor for $379!

The perfect monitor for Apple, Atari and Texas Instruments owners.

### NEW FROM SANYO — Four Great Monitors at Low CompuMart Prices.

Sanyo's new line of CRT data display monitors are specifically designed for the display of alphanumeric or graphic data.

| | |
|---|---|
| 9" Sanyo Monitor | $169 |
| 12" Sanyo Monitor | $289 |
| 12" Sanyo Monitor with green screen | $299 |
| 13" Sanyo Color Display Monitor | $495 |



COMPUMART NOW OFFERS THE ENTIRE DEC LSI-11 PRODUCT LINE. CALL FOR PRICES & DELIVERY.

## Terminals

### Televideo 912C

820 × 24—Lower case descenders. Teletype or telewriter keyboard. 110/220 VAC. 50 to 19.2K Baud Item entry pad. Great looking and no fan noise.

### Televideo 920C

Similar to the TV 912 but has programmable function keys across top. Excellent for text editing.

**Call for Introductory Sale Prices**

We've got the following Lear Siegler Terminals In Stock at prices too low to print—**Call** for quotes.

**ADM-3A** Industries favorite dumb terminal for some very smart reasons.
**ADM-3A.+** New from Lear Siegler. CALL!
**ADM-31.** The terminal that's too smart to be considered dumb.
**ADM-42.** Available with keyboard semi-intelligent terminal offering tremendous user flexibility. The optional configurations are amazing.

**Call for details.**

### HAZELTINE TERMINALS AT SPECTACULAR SAVINGS!

| | | |
|---|---|---|
| Hazeltine 1410. | List $850 | **CompuMart $749** |
| Hazeltine 1420. | List $995 | **CompuMart $895** |
| Hazeltine 1500. | List $1095 | **CompuMart $995** |
| Hazeltine 1510. | List $1395 | **CompuMart $1325** |
| Hazeltine 1520. | List $1585 | **CompuMart $1485** |
| Hazeltine 1552. | List $1395 | **CompuMart $1295** |

Call CompuMart for complete specs and quantity discounts.

## Calculators

### A CALCULATOR, A SYSTEM, A WHOLE NEW STANDARD.

### HEWLETT-PACKARD'S HP-41C

**HP-41C Calculator** ...................... **$288.00**

**The System**
**Memory Modules. For storing programs or up to 2,000 lines of program memory** ............ **$45.00**
"Extra Smart" Card Reader. Records programs and data back onto blank mag-cards ........... **$199.00**
The Printer. Upper and Lower case, High resolution plotting, Portable Thermal operation ....... **$355.00**
Application Modules ............... **$45.00 EACH**
Standard pac: Statistics, Math, Financial & Surveying

### TI CALCULATORS -

The TI Programable Calculator — **Super Sale $229.00**

### NOVATION CAT™ ACCOUSTIC MODEM

- Answer Originate
- 300 Baud
- Bell 108
- Low Profile Design

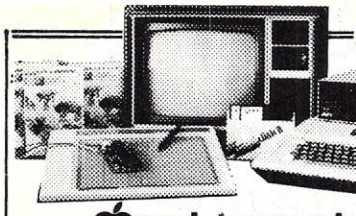
**Looks good, works great! $179.00**

### Texas Instruments TI-99/4 Home Computer

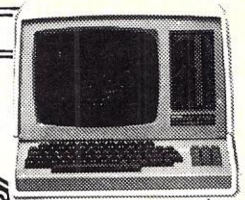
**Save $300 on this 16-Bit computer with monitor**

**TI-99/4 w/Monitor** **$889**
**TI-99/4 w/o Monitor** **$725**

# Computers



**apple computer**




## BUY-OF-THE-MONTH
## Apple III is Here

## and We've Got It!

### Brand New From Apple for the Apple II

| | |
|---|---|
| DOS 3.3 Convert disks to 16 sector format for 23% more storage and faster access | $60 |
| Apple Plot. The perfect graphic complement for Visicalc. | $70 |
| Dow Jones News & Quotes | $95 |
| Adventure | $35 |
| DOS Tool Kit | $75 |
| Apple Fortan | $200 |

### New From Videx! — Video Term.

80 column by 24 line 7 x 9 matrix — plug in compatible board for the Apple II. Price: **$325** w/o graphics EPROM

| | |
|---|---|
| Options: Graphics EPROM | $25 |
| Video Switch Plate, inserts between Apple II Video and vidoterm board | $12 |

NEW FROM MUSE FOR YOUR APPLE II

| | |
|---|---|
| The Voice | $39.95 |
| Super Text | $99.00 |
| Address Book | $49.95 |
| S.S.M. Serial & Parallel Apple Interface | $225 |
| NEW! ABT's Numeric Key pad | $110 |
| C.C.A. Data Management Program for Apple II | $ 99 |

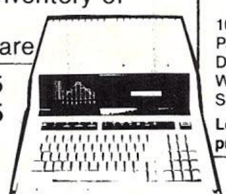
New from Mountain Hardware — Expansion Accessories for Your Apple.

| | |
|---|---|
| Introl/X-10 System | $289 |
| Super Talker | $299 |
| The Music System | $545 |
| ROM plus board w/ Keyboard filter | $199 |
| **New** from Microsoft — The Z-80 Softcard | $349 |

**Apple III Accessories**

| | |
|---|---|
| Silentype Thermal Printer | $525 |
| Disk III | $545 |
| 12″ B&W Monitor | $320 |
| Vinyl Carry Case | $ 75 |

We have a complete inventory of Apple computers, peripherals and software



### Introducing the HP-85
### $3,145

Hewlett-Packard's Personal Computer for Industry. This extremely portable computer features extended BASIC to solve your problems quickly and efficiently along with an advanced graphics system to enhance communication.

**NEW** from Hewlett-Packard
**HP 82900**-Series Flexible Disk Drives for the HP-85

These 4 new Flexible Disk Drives provide fast on-line storage using flexible disks

- HP 82901M. Supplies approx. 540K bytes of on-line storage. **$2,500**
- HP 82902M. Approx. 270K bytes of on-line storage. **$1,500**
- HP 82901S. Supplies an additional 540K bytes when connected to an HP 82901M or an HP 82902M. **$2,200**
- HP 829025. Supplies an additional 270K bytes when connected to an HP 82901M or an HP 82902M. **$1,300**

The Hewlett-Packard 7225A. High Quality/Low Cost Graphics Plotter. **$2,050**

**Call our expert sales force for complete product specifications.**

**Apple III System Packages**

Apple III Information Analyst Option A.
Includes: Apple III System with 96K RAM memory, Information analyst configuration package, 12″ B&W monitor for Apple III **$4,340**

Apple III Information Analyst Option B.
Same as Option A plus: Disk 11 for Apple III **$4,885**

Apple III Information Analyst Option C.
Same as Option A plus: Disk 11 for Apple III, Silentype Thermal Printer for Apple III **$5,410**

**Apple III Software**

| | |
|---|---|
| Visicalc III | $250 |
| Mail List Manager | $250 |
| Business Basic | $250 |
| PASCAL | $250 |
| FORTRAN | $250 |
| System Software & Manuals | $250 |

### Super Sale on Exidy Sorcerer

| | List Price | Sale! |
|---|---|---|
| 16K Sorcerer | $1,295 | $999 |
| Parallel Data Cable | $35 | $20 |
| Development PAC | $99 | $85 |
| Word Processing PAC | $199 | $169 |
| S-100 Expansion Unit | $419 | $389 |

**Lots of other Sorcerer accessories in stock at low prices — CALL!**

### ROCKWELL AIM 65

The single board development system that's perfect in the classroom or lab.

| | |
|---|---|
| Our AIM System includes: 4K AIM with BASIC interpretor assembler, Power Supply, Cassette recorder & Enclosure | $799. |
| 4K AIM — 65 | $499 |
| PL65 High Level Language | $125 |
| Paper for the AIM (roll) | $ 2.50 |
| Rockwell's 4-slot Motherboard (SALE) | $175 |
| CompuMart's Microflex 65 System for your AIM Includes: Adapter Buffer Module w/ 4-slot module stack, 8K RAM module, 16K PROM/ROM module, Asynchronous communications Interface, & Power Supply | $1,299 |

**Call or write for our complete Microflex 65 brochure**

### Zenith Z-89

The All-In-One-Computer
This is the famous computer system which takes up no more space than a terminal alone. The 12″ screen is beautiful and lends itself perfectly to professional applications thanks to its 25 lines of 80 characters. We know of no other computer which gives you this many features at such a low price.

| | |
|---|---|
| Zenith 48K Z-89 Dual Port | List $2,895 |
| **CompuMart** | **$2,695** |
| Z-19 Smart Video Terminal | List $995 |
| **CompuMart** | **$895** |

### COMMODORE

**Buy Direct from the largest Commodore dealer in the country, and the very first Commodore distributor in the U.S. Buy from the experts- Buy from CompuMart.**

Commodore—We have everything that Commodore manufactures. In stock for Immediate Delivery!
Call CompuMart now for low prices and special deals.

### ATARI 800 Personal Computer System.

**Comes with 800 Operators Manual, 16K RAM Memory module, 10K ROM Operating System, power supply, TV Switch Box.** **$1080**

PERIPHERALS

| | |
|---|---|
| Atari 410 Program Recorder (FREE w/ purchase of Atari 800) | $89.95 |
| Atari 810 Disk Drive | $699.95 |
| New Dual Disk double density | $1499.95 |
| 825 Printer (Centronics 737) | $995.00 |
| RS232 Interface w/ Cable | $249.95 |
| NEW! Light Pens | $74.95 |
| Call for New Software | |


**IMPORTANT ORDERING INFORMATION** All orders must include 4% shipping and handling. Mass. residents add 5% sales tax, Mich. residents 4% for sales tax

**TO ORDER CALL: 800-343-5504**
In Mass. call 1-617-491-2700

Phones open from 8:30 a.m. to 5:30 p.m. EST, Mon.-Fri. • P.O.'s accepted from D&B rated companies—shipment contingent upon receipt of signed purchase order • All prices are subject to change without notice • Most items in stock for immediate shipment— call for delivery quotation • In the Ann Arbor area? Our retail store is open 11:00 a.m. to 7:00 p.m. Tues.-Fri., 10:00 a.m. to 5:00 p.m. Saturdays (closed Sun. and Mon.)

# COMPUMART ✓43

270 THIRD STREET, DEPT **130** CAMBRIDGE, MA. 02142

master charge VISA

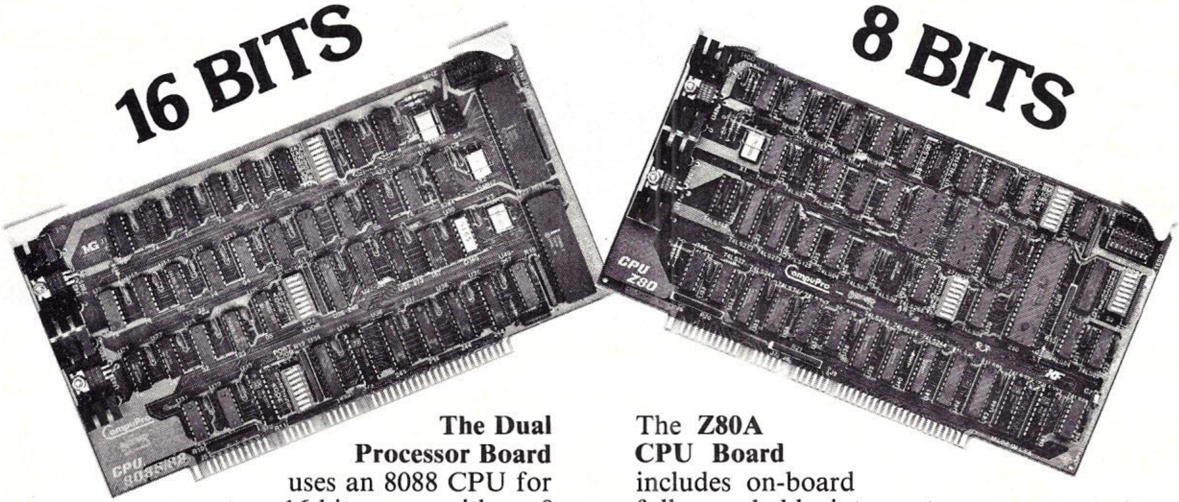
Member Computer Dealers Assoc.

We've had a reputation for dependability since 1971

✓ Reader Service index—page 241

# Throughput is the Only True Measure of Computer Performance.

**Want a 300% improvement in throughput compared to 2 MHz systems? IEEE-compatible CompuPro boards are designed *from the ground up* to operate at 6 MHz and beyond, dramatically increasing computing power and performance. Don't settle for less . . . select high speed, high reliability S-100 products from CompuPro.**




**The Dual Processor Board** uses an 8088 CPU for true 16 bit power with an 8 bit bus, and an 8085 for compatibility with CP/M and 8080 software. **SPECIAL LOW PRICES: $295 unkit, $425 assm** (both operate at 5 MHz); **$525** qualified under the high - reliability Certified System Component program (with 5 MHz 8085, 6 MHz 8088).

The **Z80A CPU Board** includes on-board fully maskable interrupts for interrupt- driven systems, provision for adding up to 8K of on-board EPROM, IEEE compatible 16/24 bit extended addressing, and much more. 4 MHz standard operation, but also works with 6 MHz Z80s. **$225 unkit, $295 assm $395 CSC.**

## HIGH SPEED S-100 MEMORY and MOTHERBOARDS

**RAM XX** (with bank select AND extended addressing) is the perfect match for either CPU board — thanks to fully static operation, extremely low power consumption, and complete IEEE spec compatibility. All **unkit** and **assembled** memories work up to 5 MHz, while Certified System Component boards run up to 8 MHz and are guaranteed to work with 6 MHz Z80s. All CompuPro motherboards work up to 10 MHz.

| | unkit | assm | CSC |
|---|---|---|---|
| **16K RAM XX-16** | **$349** | **$419** | **$519** |
| **24K RAM XX-24** | **$479** | **$539** | **$649** |
| **32K RAM XX-32** | **$649** | **$699** | **$799** |
| **20 slot motherboard with edge connectors** | **$174** | **$214** | n/a |
| **12 slot motherboard with edge connectors** | **$129** | **$169** | n/a |
| **6 slot motherboard with edge connectors** | **$89** | **$129** | n/a |

**SEE COMPUPRO PRODUCTS IN PERSON AT COMPUTER STORES WORLD-WIDE, OR WRITE US DIRECT IF THERE'S NO STORE IN YOUR AREA.**

**TERMS:** Cal res add tax. Allow 5% for shipping, excess refunded. VISA®/Mastercard® orders call **(415) 562-0636**, 24 hours. Please include street address for UPS delivery. Sale prices good through cover month of magazine, other prices subject to change without notice.

✓42

**CompuPro**™ from GODBOUT ELECTRONICS

**Bldg. 725, Oakland Airport, CA 94614**

NEW!

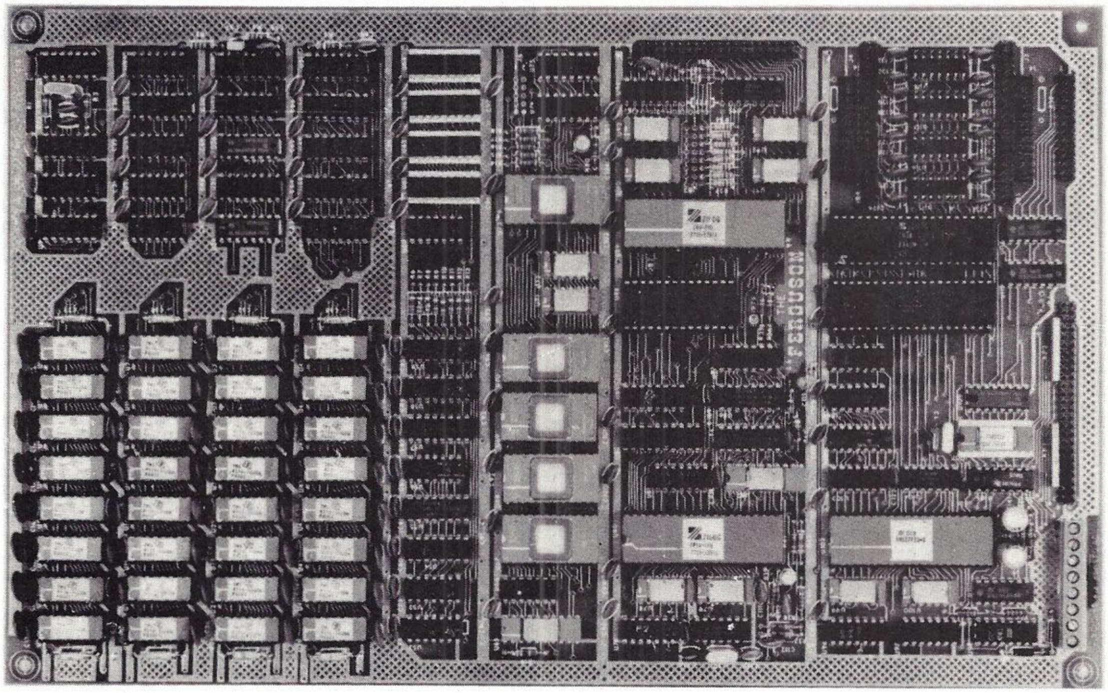
"THE BIG BOARD"

OEM - INDUSTRIAL - BUSINESS - SCIENTIFIC

# SINGLE BOARD COMPUTER KIT!

## Z-80 CPU! 64K RAM!

NEW!



**THE FERGUSON PROJECT:** Three years in the works, and maybe too good to be true. A tribute to hard headed, no compromise, high performance, American engineering! The Big Board gives you all the most needed computing features on one board at a very reasonable cost. The Big Board was designed from scratch to run the latest version of CP/M*. Just imagine all the off-the-shelf software that can be run on the Big Board without any modifications needed! Take a Big Board, add a couple of 8 inch disc drives, power supply, and an enclosure; and you have a total Business System for about 1/3 the cost you might expect to pay.

**$649.00 **** (64K KIT BASIC I/O)

SIZE: 8½ x 13¾ IN.
SAME AS AN 8 IN. DRIVE.
REQUIRES: +5V @ 3 AMPS
+ - 12V @ .5 AMPS.

## FEATURES: (Remember, all this on one board!)

### 64K RAM

Uses industry standard 4116 RAM'S. All 64K is available to the user, our VIDEO and EPROM sections do not make holes in system RAM. Also, very special care was taken in the RAM array PC layout to eliminate potential noise and glitches.

### Z-80 CPU

Running at 2.5 MHZ. Handles all 4116 RAM refresh and supports Mode 2 INTERUPTS. Fully buffered and runs 8080 software.

### SERIAL I/O (OPTIONAL)

Full 2 channels using the Z80 SIO and the SMC 8116 Baud Rate Generator. FULL RS232! For synchronous or asynchronous communication. In synchronous mode, the clocks can be transmitted or received by a modem. Both channels can be set up for either data-communication or data-terminals. Supports mode 2 Int. Price for all parts and connectors: $85.

### BASIC I/O

Consists of a separate parallel port (Z80 PIO) for use with an ASCII encoded keyboard for input. Output would be on the 80 x 24 Video Display.

### 80 x 24 CHARACTER VIDEO

With a crisp, flicker-free display that looks extremely sharp even on small monitors. Hardware scroll and full cursor control. Composite video or split video and sync. Character set is supplied on a 2716 style ROM, making customized fonts easy. Sync pulses can be any desired length or polarity. Video may be inverted or true.

### FLOPPY DISC CONTROLLER

Uses WD1771 controller chip with a TTL Data Separator for enhanced reliability. IBM 3740 compatible. Supports up to four 8 inch disc drives. Directly compatible with standard Shugart drives such as the SA800 or SA801. Drives can be configured for remote AC off-on. Runs CP/M* 2.2.

### FOUR PORT PARALLEL I/O (OPTIONAL)

Uses Z-80 PIO. Full 16 bits, fully buffered, bi-directional. User selectable hand shake polarity. Set of all parts and connectors for parallel I/O: $29.95

### REAL TIME CLOCK (OPTIONAL)

Uses Z-80 CTC. Can be configured as a Counter on Real Time Clock. Set of all parts: $14.95

**SYSTEM COMPARISON**

| | |
|---|---|
| 64K RAM KIT | $370.00 |
| 80 x 24 Video Kit | 365.00 |
| Floppy Disk Controller Kit | 235.00 |
| Z-80 CPU Kit | 185.95 |
| SER & PAR. I/O | 129.95 |
| S-100 Mother Board | 45.00 |
| SUB TOTAL | $1330.90 |

Talk about bangs per buck! The prices shown for S100 kits were taken from the July 1980 BYTE. This will give some basis for comparison between the Big Board and a similar system implementation on the S100 Buss.

### CP/M* 2.2 FOR BIG BOARD

The popular CP/M* D.O.S. modified by MICRONIX SYSTEMS to run on Big Board is available for $150.00.

## FIRST TIME OFFERED!

### PFM 3.0 2K SYSTEM MONITOR

The real power of the Big Board lies in its PFM 3.0 on board monitor. PFM commands include: Dump Memory, Boot CP/M*, Copy, Examine, Fill Memory, Test Memory, Go To, Read and Write I/O Ports, Disc Read (Drive, Track, Sector), and Search. PFM occupies one of the four 2716 EPROM locations provided. It does not occupy any of the 64K of system RAM!

# Digital Research Computers

(OF TEXAS)

P.O. BOX 401565 • GARLAND, TEXAS 75040 • (214) 271-3538

**TERMS:** Initial shipments will be made approximately 3 to 5 weeks after we receive your order. VISA, MC, cash accepted. We will accept COD's (for the Big Board only) with a $75 deposit. Balance UPS COD. The $75 deposit assures your place in line for the initial production run of Big Board.

*TRADEMARK OF DIGITAL RESEARCH. NOT ASSOCIATED WITH DIGITAL RESEARCH OF CALIFORNIA, THE SUPPLIERS OF CPM SOFTWARE

** 1 TO 4 PIECE DOMESTIC USA PRICE.

# 7400 TTL

| | |
|---|---|
| SN7400N | .25 |
| SN7401N | .20 |
| SN7402N | .25 |
| SN7403N | .25 |
| SN7404N | .25 |
| SN7405N | .25 |
| SN7406N | .35 |
| SN7407N | .35 |
| SN7408N | .25 |
| SN7409N | .25 |
| SN7410N | .25 |
| SN7411N | .25 |
| SN7412N | .35 |
| SN7413N | .40 |
| SN7414N | .85 |
| SN7416N | .29 |
| SN7417N | .29 |
| SN7420N | .20 |
| SN7421N | .29 |
| SN7422N | .39 |
| SN7423N | .25 |
| SN7425N | .29 |
| SN7426N | .29 |
| SN7427N | .25 |
| SN7429N | .39 |
| SN7430N | .25 |
| SN7432N | .25 |
| SN7437N | .25 |
| SN7438N | .40 |
| SN7439N | .25 |
| SN7440N | .20 |
| SN7441N | .89 |
| SN7442N | .59 |
| SN7443N | .75 |
| SN7444N | .75 |
| SN7445N | .75 |
| SN7446N | .79 |
| SN7447N | .69 |
| SN7448N | .79 |
| SN7450N | .20 |
| SN7451N | .20 |
| SN7453N | .20 |
| SN7454N | .20 |
| SN7459A | .25 |
| SN7460N | .20 |

| | |
|---|---|
| SN7470N | .29 |
| SN7472N | .29 |
| SN7473N | .35 |
| SN7474N | .35 |
| SN7475N | .49 |
| SN7476N | .35 |
| SN7479N | 5.00 |
| SN7480N | .50 |
| SN7482N | .99 |
| SN7483N | .69 |
| SN7485N | .89 |
| SN7486N | .35 |
| SN7489N | 1.75 |
| SN7490N | .49 |
| SN7491N | .59 |
| SN7492N | .43 |
| SN7493N | .43 |
| SN7494N | .65 |
| SN7495N | .65 |
| SN7496N | .65 |
| SN7497N | 3.00 |
| SN74100N | 1.49 |
| SN74107N | .35 |
| SN74109N | .39 |
| SN74116N | 1.95 |
| SN74121N | .39 |
| SN74122N | .39 |
| SN74123N | .59 |
| SN74125N | .49 |
| SN74126N | .49 |
| SN74132N | .75 |
| SN74136N | .75 |
| SN74141N | .79 |
| SN74142N | 2.95 |
| SN74143N | 2.95 |
| SN74144N | 2.95 |
| SN74145N | .79 |
| SN74147N | 1.95 |
| SN74148N | 1.29 |
| SN74150N | 1.25 |
| SN74151N | .59 |
| SN74152N | .59 |
| SN74153N | .79 |
| SN74154N | 1.25 |
| SN74155N | .79 |
| SN74156N | .79 |
| SN74157N | .65 |

| | |
|---|---|
| SN74160N | .89 |
| SN74161N | .89 |
| SN74162N | .89 |
| SN74163N | .89 |
| SN74164N | .89 |
| SN74165N | .89 |
| SN74166N | 1.25 |
| SN74167N | 1.95 |
| SN74170N | 1.95 |
| SN74172N | 6.00 |
| SN74173N | 1.25 |
| SN74174N | .99 |
| SN74175N | .89 |
| SN74176N | .79 |
| SN74177N | .79 |
| SN74179N | 1.95 |
| SN74180N | .79 |
| SN74181N | 1.95 |
| SN74182N | .79 |
| SN74184N | 2.25 |
| SN74185N | 2.25 |
| SN74186N | 9.95 |
| SN74188N | 3.95 |
| SN74190N | 1.25 |
| SN74191N | 1.25 |
| SN74192N | .89 |
| SN74193N | .89 |
| SN74194N | .89 |
| SN74195N | .69 |
| SN74196N | .89 |
| SN74197N | .89 |
| SN74198N | 1.49 |
| SN74199N | 1.49 |
| SN74S200 | 4.95 |
| SN74251N | .99 |
| SN74279N | .79 |
| SN74283N | 2.25 |
| SN74284N | 3.95 |
| SN74285N | 3.95 |
| SN74365N | .69 |
| SN74366N | .69 |
| SN74367N | .69 |
| SN74368N | .69 |
| SN74390N | 1.95 |
| SN74393N | 1.95 |

# CMOS

| | |
|---|---|
| CD4000 | .39 |
| CD4001 | .39 |
| CD4002 | .39 |
| CD4006 | 1.19 |
| CD4007 | .25 |
| CD4009 | .49 |
| CD4010 | .49 |
| CD4011 | .39 |
| CD4012 | .25 |
| CD4013 | .49 |
| CD4014 | 1.39 |
| CD4015 | 1.19 |
| CD4016 | .59 |
| CD4017 | 1.19 |
| CD4018 | .99 |
| CD4019 | .49 |
| CD4020 | 1.19 |
| CD4021 | 1.39 |
| CD4022 | 1.19 |
| CD4023 | .29 |
| CD4024 | .79 |
| CD4025 | .23 |
| CD4026 | 2.95 |
| CD4027 | .69 |

| | |
|---|---|
| CD4028 | .89 |
| CD4029 | 1.49 |
| CD4030 | .49 |
| CD4035 | .99 |
| CD4040 | 1.49 |
| CD4041 | 1.49 |
| CD4042 | .99 |
| CD4043 | .89 |
| CD4044 | .89 |
| CD4046 | 1.79 |
| CD4047 | 2.50 |
| CD4048 | 1.35 |
| CD4049 | .49 |
| CD4050 | .69 |
| CD4051 | 1.19 |
| CD4053 | 1.19 |
| CD4056 | 2.95 |
| CD4059 | 9.95 |
| CD4060 | 1.49 |
| CD4066 | .79 |
| CD4068 | .39 |
| CD4069 | .45 |

| | |
|---|---|
| CD4070 | .55 |
| CD4071 | .49 |
| CD4072 | .49 |
| CD4076 | 1.39 |
| CD4081 | .39 |
| CD4082 | .39 |
| CD4093 | .99 |
| CD4098 | 1.19 |
| MC14409 | 14.95 |
| MC14410 | 14.95 |
| MC14411 | 14.95 |
| MC14419 | 4.95 |
| MC14433 | 13.95 |
| MC14506 | .75 |
| MC14507 | .99 |
| MC14562 | 11.95 |
| MC14583 | 2.49 |
| CD4508 | 3.95 |
| CD4510 | 1.39 |
| CD4511 | 1.29 |
| CD4515 | 2.95 |
| CD4518 | 1.79 |
| CD4520 | 1.29 |
| CD4566 | 2.25 |

# 74C00

| | |
|---|---|
| 74C00 | .39 |
| 74C02 | .39 |
| 74C04 | .39 |
| 74C08 | .39 |
| 74C10 | .39 |
| 74C14 | .75 |
| 74C20 | .39 |
| 74C30 | .39 |
| 74C42 | 1.39 |
| 74C48 | 1.95 |
| 74C73 | .79 |
| 74C74 | .79 |

| | |
|---|---|
| 74C85 | 1.95 |
| 74C90 | 1.29 |
| 74C93 | 1.29 |
| 74C95 | 1.59 |
| 74C107 | 1.89 |
| 74C151 | 2.95 |
| 74C154 | 3.95 |
| 74C157 | 2.25 |
| 74C160 | 1.69 |
| 74C161 | 1.69 |

| | |
|---|---|
| 74C163 | 1.69 |
| 74C164 | 1.59 |
| 74C173 | 1.39 |
| 74C192 | 1.69 |
| 74C193 | 1.69 |
| 74C195 | 1.59 |
| 74C922 | 5.49 |
| 74C923 | 5.75 |
| 74C925 | 7.50 |
| 74C926 | 7.50 |
| 80C95 | .79 |
| 80C97 | .79 |

# LINEAR

| | |
|---|---|
| 78MG | 1.75 |
| LM106H | .99 |
| LM300H | .80 |
| LM301CN/H | .35 |
| LM302H | 1.75 |
| LM304H | 1.49 |
| LM305H | .79 |
| LM307CN/H | .45 |
| LM308CN/H | 1.00 |
| LM309H | 1.10 |
| LM309K | 1.25 |
| LM310CN | 1.95 |
| LM311N/H | .90 |
| LM312H | 1.95 |
| LM317K | 3.95 |
| LM318CN/H | 1.95 |
| LM319N | 1.30 |
| LM320K-5 | 1.35 |
| LM320K-5.2 | 1.35 |
| LM320K-12 | 1.25 |
| LM320K-15 | 1.35 |
| LM320K-18 | 1.35 |
| LM320K-24 | 1.35 |
| LM320T-5 | 1.25 |
| LM320T-5.2 | 1.25 |
| LM320T-8 | 1.25 |
| LM320T-12 | 1.25 |
| LM320T-15 | 1.25 |
| LM320T-18 | 1.25 |
| LM320T-24 | 1.25 |
| LM323K-5 | 5.95 |
| LM324N | .99 |
| LM339N | .99 |
| LM340K-5 | 1.35 |
| LM340K-6 | 1.35 |
| LM340K-8 | 1.35 |
| LM340K-12 | 1.35 |
| LM340K-15 | 1.35 |

| | |
|---|---|
| LM340K-18 | 1.35 |
| LM340K-24 | 1.35 |
| LM340T-5 | 1.25 |
| LM340T-6 | 1.25 |
| LM340T-8 | 1.25 |
| LM340T-12 | 1.25 |
| LM340T-15 | 1.25 |
| LM340T-18 | 1.25 |
| LM340T-24 | 1.25 |
| LM358N | 1.00 |
| LM370N | 1.95 |
| LM373N | 3.25 |
| LM377N | 2.95 |
| LM380N | 1.25 |
| LM380CN | .99 |
| LM381N | 1.79 |
| LM382N | 1.79 |
| NE510A | 6.00 |
| NE529A | 4.95 |
| NE531H/V | 3.95 |
| NE536T | 6.00 |
| NE540 | 6.00 |
| NE544N | 4.95 |
| NE550N | 1.30 |
| NE555V | .39 |
| NE556N | .99 |
| NE560B | 5.00 |
| NE562B | 5.00 |
| LM565N/H | 1.25 |
| LM566CN | 1.75 |
| LM567V/H | .99 |
| NE570N | 4.95 |
| LM703CN/H | .69 |
| LM709N/H | .29 |

| | |
|---|---|
| LM710N | .79 |
| LM711N | .39 |
| LM723N/H | .55 |
| LM733N | 1.00 |
| LM739N | 1.19 |
| LM741CN/H | .35 |
| LM741-14N | .39 |
| LM747N/H | .79 |
| LM748N/H | .59 |
| LM1310N | 1.95 |
| LM1458CN/H | .59 |
| MC1488N | 1.95 |
| MC1489N | 1.95 |
| LM1496N | .95 |
| LM1556V | 1.75 |
| MC1741SCP | 3.00 |
| LM2111N | 1.95 |
| LM2901N | 1.95 |
| LM3053N | 1.50 |
| LM3065N | 1.49 |
| LM3900N(3401) | .59 |
| LM3905N | 1.49 |
| LM3909N | 1.25 |
| MC5558V | .59 |
| 8038B | 4.95 |
| LM75450N | .49 |
| 75451CN | .39 |
| 75452CN | .39 |
| 75453CN | .39 |
| 75454CN | .39 |
| 75491CN | .79 |
| 75492CN | .89 |
| 75493N | 1.25 |
| 75494CN | 1.25 |
| RC4136 | 1.25 |
| RC4151 | 3.95 |
| RC4194 | 4.95 |
| RC4195 | 4.49 |

# 74LS00TTL

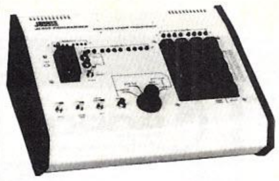
| | |
|---|---|
| 74LS00 | .35 |
| 74LS01 | .35 |
| 74LS02 | .35 |
| 74LS03 | .35 |
| 74LS04 | .42 |
| 74LS05 | .42 |
| 74LS08 | .35 |
| 74LS09 | .42 |
| 74LS10 | .35 |
| 74LS11 | .75 |
| 74LS13 | .59 |
| 74LS14 | 1.25 |
| 74LS15 | .35 |
| 74LS20 | .35 |
| 74LS21 | .35 |
| 74LS22 | .35 |
| 74LS26 | .35 |
| 74LS27 | .35 |
| 74LS28 | .35 |
| 74LS30 | .35 |
| 74LS32 | .42 |
| 74LS37 | .45 |
| 74LS40 | .35 |
| 74LS42 | 1.05 |
| 74LS47 | 1.05 |

| | |
|---|---|
| 74LS51 | .29 |
| 74LS54 | .29 |
| 74LS55 | .29 |
| 74LS73 | .54 |
| 74LS74 | .54 |
| 74LS75 | .71 |
| 74LS76 | .54 |
| 74LS78 | .49 |
| 74LS83 | 1.05 |
| 74LS85 | 1.50 |
| 74LS86 | .54 |
| 74LS90 | .71 |
| 74LS92 | .90 |
| 74LS93 | .90 |
| 74LS95 | .99 |
| 74LS96 | 1.15 |
| 74LS107 | .54 |
| 74LS109 | .54 |
| 74LS112 | .54 |
| 74LS123 | 1.50 |
| 74LS125 | 1.05 |
| 74LS132 | .99 |
| 74LS136 | .59 |
| 74LS138 | 1.05 |

| | |
|---|---|
| 74LS139 | 1.05 |
| 74LS151 | 1.05 |
| 74LS155 | 1.05 |
| 74LS157 | 1.05 |
| 74LS160 | 1.15 |
| 74LS161 | 1.39 |
| 74LS162 | 1.25 |
| 74LS163 | 1.39 |
| 74LS164 | 1.50 |
| 74LS175 | 1.25 |
| 74LS181 | 2.49 |
| 74LS190 | 1.25 |
| 74LS191 | 1.25 |
| 74LS192 | 1.39 |
| 74LS193 | 1.39 |
| 74LS194 | 1.39 |
| 74LS195 | 1.39 |
| 74LS253 | 1.25 |
| 74LS257 | 1.05 |
| 74LS258 | 1.75 |
| 74LS260 | .83 |
| 74LS279 | .90 |
| 74LS367 | .75 |
| 74LS368 | .75 |
| 74LS670 | 2.49 |

# JE608 PROGRAMMER

## 2704/2708 EPROM PROGRAMMER



- 3 separate Display Registers: 8 LED's for Hex Key entries, 10 LED's (2°-2°) for Address Register and 8 LED's for Data Memory Register. The Data Memory Register displays the content of the RAMs from the EPROM Chip.
- Development of microprocessor systems by means of a ribbon cable from the programmer panel test socket to the EPROM socket on the microprocessor board.
- Rapid checking verification of programmed data changes.
- User may move data from a master to RAM's or write into RAM's with keyboard entries.
- Allows manual stepping manipulation (up and down) at any address location.
- Stand-alone EPROM Programmer consisting of: A 19-key Hexadecimal Keyboard assembly, Programmer Board assembly with 4 power supplies and a LED/Test Socket Panel Board assembly. The Test Socket is zero force insertion type. Power requirements: 115VAC, 60Hz, 6W.
- Compact desk-top enclosure: Color-coordinated designer's case with light tan panels and molded end pieces in mocha brown. Size: 3¾"H x 11"W x 8¾"D. Weight: 5 lbs.

The JE608 EPROM Programmer is a completely self-contained unit which is independent of computer control and requires no additional systems for its operations. The EPROM can be programmed from the Hexadecimal Keyboard or from a pre-programmed EPROM. The JE608 Programmer can emulate a programmed EPROM by the use of its internal RAM circuits. This will allow the user to test or pretest a program, for a system, prior to programming a chip. Any changes in the program can be entered directly into the memory circuits with the Hexadecimal Keyboard so that rewriting the entire program will not be necessary. The JE608 Programmer contains a Programmer/Board with 25 IC's and including power supplies of: −5V, +5V, +12V and +26V. The Hexadecimal Keyboard and LED/Test Socket Panel Board are separate assemblies within the system.

**JE608K KIT** ............................................ **$399.95**
**JE608A Assembled and tested** ........................ **$499.95**

# DISCRETE LEDS

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| XC556R | .200" | red | 5/$1 | MV50 | .085" | red | 6/$1 | XC111R | .190" | red | 5/$1 |
| XC556G | .200" | green | 4/$1 | XC209R | .125" | red | 5/$1 | XC111G | .190" | green | 4/$1 |
| XC556Y | .200" | yellow | 4/$1 | XC209G | .125" | green | 4/$1 | XC111Y | .190" | yellow | 4/$1 |
| XC556C | .200" | clear | 4/$1 | XC209Y | .125" | yellow | 4/$1 | XC111C | .190" | clear | 4/$1 |
| XC22R | .200" | red | 5/$1 | XC526R | .185" | red | 5/$1 | | | | |
| XC22G | .200" | green | 4/$1 | XC526G | .185" | green | 4/$1 | | | | |
| XC22Y | .200" | yellow | 4/$1 | XC526Y | .185" | yellow | 4/$1 | | | | |
| MV10B | .170" | red | 4/$1 | XC526C | .185" | clear | 4/$1 | | | | |

INFRA-RED LED
¼"x ¼"x 1/16" flat
IRL — 5/$1

# DISPLAY LEDS

C.A. — Common Anode    C.C. — Common Cathode

| Type | Polarity | Ht | Price | Type | Polarity | Ht | Price |
|---|---|---|---|---|---|---|---|
| MAN 1 | C.A.—red | .270 | 2.95 | DL707 | C.A.—red | .300 | .99 |
| MAN 2 | 5x7 D.M.—red | .300 | 4.95 | DL728 | C.C.—red | .500 | 1.49 |
| MAN 3 | C.C.—red | .125 | .25 | DL741 | C.A.—red | .600 | 1.25 |
| MAN 4 | C.C.—red | .187 | 1.95 | DL746 | C.A.—red ± 1 | .630 | 1.49 |
| MAN 7G | C.A.—green | .300 | 1.25 | DL747 | C.A.—red | .600 | 1.49 |
| MAN 7Y | C.A.—yellow | .300 | .99 | DL750 | C.C.—red | .600 | 1.49 |
| MAN 72 | C.A.—red | .300 | .75 | DL33B | C.C.—red | .110 | .35 |
| MAN 74 | C.C.—red | .300 | 1.25 | FND70 | C.C. | .250 | .69 |
| MAN 82 | C.A.—yellow | .300 | .49 | FND358 | C.C. ± 1 | .357 | .99 |
| MAN 84 | C.C.—yellow | .300 | .99 | FND359 | C.C. | .357 | .75 |
| MAN 3620 | C.A.—orange | .300 | .49 | FND503 | C.C. (FND500) | .500 | .99 |
| MAN 3630 | C.A.—orange ± 1 | .300 | .99 | FND507 | C.A. (FND510) | .500 | .99 |
| MAN 3640 | C.C.—orange | .300 | .99 | 5082-7730 | C.A.—red | .300 | .99 |
| MAN 4610 | C.A.—orange | .300 | .99 | HDSP-3400 | C.A.—red | .800 | 1.50 |
| MAN 4810 | C.A.—yellow | .400 | .99 | HDSP-3403 | C.C.—red | .800 | 1.50 |
| MAN 4840 | C.C.—yellow | .400 | .99 | 5082-7613 | C.C., R.H.D.—red | .300 | 1.25 |
| MAN 6610 | C.A.—orange—DD | .560 | .99 | 5082-7620 | C.A., L.H.D.—yel. | .300 | 1.25 |
| MAN 6630 | C.A.—orange ± 1 | .560 | .99 | 5082-7623 | C.C., R.H.D.—yel. | .300 | 1.25 |
| MAN 6640 | C.C.—orange—DD | .560 | .99 | 5082-7730 | C.A., L.H.D.—red | .300 | .99 |
| MAN 6650 | C.C.—orange ± 1 | .560 | .99 | 5082-7731 | C.A., R.H.D.—red | .300 | 1.25 |
| MAN 6660 | C.A.—orange | .560 | .99 | 5082-7750 | C.A., L.H.D.—red | .430 | 1.25 |
| MAN 6710 | C.A.—red—DD | .560 | .99 | 5082-7751 | C.A., R.H.D.—red | .430 | 1.25 |
| MAN 6750 | C.C.—red ± 1 | .560 | .99 | 5082-7760 | C.C., R.H.D.—red | .430 | 1.25 |
| MAN 6760 | C.A.—red | .560 | .99 | 5082-7300 | 4x7 sgl. dig. RHDP | .600 | 19.95 |
| MAN 6780 | C.C.—red | .560 | .99 | 5082-7302 | 4x7 sgl. dig. LHDP | .600 | 19.95 |
| DL701 | C.A.—red ± 1 | .300 | .99 | 5082-7304 | Overnge. char. (±1) | .600 | 15.00 |
| DL704 | C.C.—red | .300 | .99 | 5082-7340 | 4x7 sgl. dig. hex. | .600 | 22.50 |

# RCA LINEAR

| | | | |
|---|---|---|---|
| CA3013T | 2.15 | CA3082N | 2.00 |
| CA2023T | 3.25 | CA3083N | 1.60 |
| CA3035T | 2.48 | CA3086N | .85 |
| CA3039T | 1.35 | CA3089N | 3.75 |
| CA3046N | 1.30 | CA3130T | 1.39 |
| CA3059N | 3.25 | CA3140T | 1.25 |
| CA3060N | 3.25 | CA3160T | 1.25 |
| CA3080T | 1.25 | CA3401N | .59 |
| CA3081N | 2.00 | CA3600N | 3.50 |

## CALCULATOR CHIPS/DRIVERS

| | |
|---|---|
| MM5725 | $2.95 |
| MM5738 | 2.95 |
| DM8864 | 2.00 |
| DM8865 | 1.00 |
| DM8887 | .75 |
| DM8889 | .75 |
| 9374 7-seg. LED driver | 1.50 |
| MM5309 | 4.95 |

## CLOCK CHIPS

| | |
|---|---|
| MM5309 | 4.95 |
| MM5311 | 4.95 |
| MM5312 | 4.95 |
| MM5314 | 4.95 |
| MM5316 | 6.95 |
| MM5318 | 9.95 |
| MM5369 | 2.95 |
| MM5377 | 4.95 |
| MM5387/1998a | 4.95 |

## MOTOROLA

| | |
|---|---|
| MC1408L7 | 4.95 |
| MC1408L8 | 5.75 |
| MC1439L | 2.95 |
| MC3022P | 2.95 |
| MC3061P | 3.50 |
| MC4016(74416) | 7.50 |
| MC4024P | 3.95 |
| MC4040P | 6.95 |
| MC4044P | 4.50 |

## LOW PROFILE (TIN) SOCKETS

| | 1-24 | 25-49 | 50-100 |
|---|---|---|---|
| 8 pin LP | .17 | .16 | .15 |
| 14 pin LP | .20 | .19 | .18 |
| 16 pin LP | .22 | .21 | .20 |
| 18 pin LP | .29 | .28 | .27 |
| 20 pin LP | .34 | .32 | .30 |
| 22 pin LP | .37 | .36 | .35 |
| 24 pin LP | .38 | .37 | .36 |
| 28 pin LP | .45 | .44 | .43 |
| 36 pin LP | .60 | .59 | .58 |
| 40 pin LP | .63 | .62 | .61 |

## SOLDERTAIL STANDARD (TIN)

| | 1-24 | 25-49 | 50-100 |
|---|---|---|---|
| 14 pin ST | .27 | .25 | .24 |
| 16 pin ST | .30 | .27 | .25 |
| 18 pin ST | .35 | .32 | .30 |
| 24 pin ST | .49 | .45 | .42 |
| 28 pin ST | .99 | .90 | .81 |
| 36 pin ST | 1.39 | 1.26 | 1.15 |
| 40 pin ST | 1.59 | 1.45 | 1.30 |

## SOLDERTAIL (GOLD) STANDARD

| | 1-24 | 25-49 | 50-100 |
|---|---|---|---|
| 8 pin SG | .39 | .35 | .31 |
| 14 pin SG | .49 | .45 | .41 |
| 16 pin SG | .54 | .49 | .44 |
| 18 pin SG | .59 | .53 | .48 |
| 24 pin SG | .79 | .75 | .69 |
| 28 pin SG | 1.10 | 1.00 | .90 |
| 36 pin SG | 1.65 | 1.40 | 1.26 |
| 40 pin SG | 1.75 | 1.59 | 1.45 |

## WIRE WRAP SOCKETS (GOLD) LEVEL #3

| | 1-24 | 25-49 | 50-100 |
|---|---|---|---|
| 8 pin WW | .59 | .54 | .49 |
| 10 pin WW | .69 | .63 | .58 |
| 14 pin WW | .79 | .73 | .67 |
| 16 pin WW | .85 | .77 | .70 |
| 18 pin WW | .99 | .90 | .81 |
| 20 pin WW | 1.19 | 1.08 | .99 |
| 22 pin WW | 1.49 | 1.35 | 1.23 |
| 24 pin WW | 1.39 | 1.26 | 1.14 |
| 28 pin WW | 1.69 | 1.53 | 1.38 |
| 36 pin WW | 2.19 | 1.99 | 1.79 |
| 40 pin WW | 2.29 | 2.09 | 1.89 |

# 1/4 WATT RESISTOR ASSORTMENTS – 5%

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| ASST. 1 | 5 ea. | 10 OHM / 27 OHM | 12 OHM / 33 OHM | 15 OHM / 39 OHM | 18 OHM / 47 OHM | 22 OHM / 56 OHM | 50 PCS | $1.95 |
| ASST. 2 | 5 ea. | 68 OHM / 180 OHM | 82 OHM / 220 OHM | 100 OHM / 270 OHM | 120 OHM / 330 OHM | 150 OHM / 390 OHM | 50 PCS | $1.95 |
| ASST. 3 | 5 ea. | 470 OHM / 1.2K | 560 OHM / 1.5K | 680 OHM / 1.8K | 820 OHM / 2.2K | 1K / 2.7K | 50 PCS | $1.95 |
| ASST. 4 | 5 ea. | 3.3K / 8.2K | 3.9K / 10K | 4.7K / 12K | 5.6K / 15K | 6.8K / 18K | 50 PCS | $1.95 |
| ASST. 5 | 5 ea. | 22K / 56K | 27K / 68K | 33K / 82K | 39K / 100K | 47K / 120K | 50 PCS | $1.95 |
| ASST. 6 | 5 ea. | 150K / 390K | 180K / 470K | 220K / 560K | 270K / 680K | 330K / 820K | 50 PCS | $1.95 |
| ASST. 7 | 5 ea. | 1M / 2.7M | 1.2M / 3.3M | 1.5M / 3.9M | 1.8M / 4.7M | 2.2M / 5.6M | 50 PCS | $1.95 |

**ASST. 8R** Includes Resistor Assortments 1-7 (350 PCS.) **$10.95 ea.**

$10.00 Min. Order — U.S. Funds Only
Calif. Residents Add 6% Sales Tax
Postage — Add 5% plus $1 Insurance (if desired)

Spec Sheets — 25¢
1981 Catalog Available — Send 41¢ stamp

NEW 1981 CATALOG

**Jameco ELECTRONICS**

PHONE ORDERS WELCOME
(415) 592-8097

*MAIL ORDER ELECTRONICS — WORLDWIDE*
1355 SHOREWAY ROAD, BELMONT, CA 94002 ✓41
10/80 PRICES SUBJECT TO CHANGE

# TELEPHONE/KEYBOARD CHIPS

| | | |
|---|---|---|
| AY-5-9100 | Push Button Telephone Dialler | $14.95 |
| AY-5-9200 | Repertory Dialler | 14.95 |
| AY-5-9500 | CMOS Clock Generator | 4.95 |
| AY-5-2376 | Keyboard Encoder (88 keys) | 11.95 |
| HD0165 | Keyboard Encoder (16 keys) | 7.95 |
| 74C922 | Keyboard Encoder (16 keys) | 5.49 |
| 74C923 | Keyboard Encoder (20 keys) | 5.75 |

# ICM CHIPS

| | | |
|---|---|---|
| ICM7045 | CMOS Precision Timer | 24.95 |
| ICM7205 | CMOS LED Stopwatch/Timer | 19.95 |
| ICM7207 | Oscillator Controller | 7.50 |
| ICM7208 | Seven Decade Counter | 19.95 |
| ICM7209 | Clock Generator | 6.95 |

# NMOS READ ONLY MEMORIES

| | | |
|---|---|---|
| MCM6571 | 128 X 9 X 7 ASCII Shifted with Greek | 13.50 |
| MCM6574 | 128 X 9 X 7 Math Symbol & Pictures | 13.50 |
| MCM6575 | 128 X 9 X 7 Alpha Control Char. Gen | 13.50 |

# MISCELLANEOUS

| | | |
|---|---|---|
| TL074CN | Quad Low Noise bi-fet Op Amp | 2.49 |
| TL494CN | Switching Regulator | 4.49 |
| TL496CP | Single Switching Regulator | 1.75 |
| 11C90 | Divide 10/11 Prescaler | 14.95 |
| 95H90 | Hi-Speed Divide 10/11 Prescaler | 11.95 |
| 4N33 | Photo-Darlington Opto-Isolator | 1.25 |
| MK50240 | Top Octave Freq. Generator | 17.50 |
| DS0026CH | 5Mhz 2-phase MOS clock driver | 3.75 |
| TIL308 | .27" red num. display w/integ. logic chip | 10.95 |
| MM5320 | TV Camera Sync. Generator | 14.95 |
| MM5330 | 4½ Digit DPM Logic Block **(Special)** | 3.95 |
| LD110/111 | 3½ Digit A/D Converter Set | 25.00/set |
| MC14433P | 3½ Digit A/D Converter | 13.95 |

**LITRONIX ISO-LIT 1**
Photo Transistor Opto-Isolator
(Same as MCT 2 or 4N25)
**49¢ each**

**SN 76477**
SOUND GENERATOR
Generates Complex Sounds
Low Power - Programmable
**$3.95 each**

**TV GAME CHIP AND CRYSTAL**
AY-3-8500-1 and 2.01 MHZ Crystal (Chip & Crystal includes score display, 6 games and select angles, etc. **7.95/set**

# EXAR

| | | | | | |
|---|---|---|---|---|---|
| XR205 | $8.40 | | | XR2242CP | 1.50 |
| XR210 | 4.40 | | | XR2264 | 4.25 |
| XR215 | 4.40 | | | XR2556 | 3.20 |
| XR320 | 1.55 | | | XR2567 | 2.99 |
| XR-L555 | 1.50 | JE2206KB | 19.95 | XR3403 | 1.25 |
| XR555 | .39 | XR1800 | 3.20 | XR4136 | 1.25 |
| XR556 | .99 | XR2206 | 4.40 | XR4151 | 3.95 |
| XR567CP | .99 | XR2207 | 3.85 | XR4194 | 1.45 |
| XR567CT | 1.25 | XR2208 | 5.20 | XR4202 | 3.60 |
| XR1310P | 1.95 | XR2209 | 1.75 | XR4212 | 2.05 |
| XR1468CN | 3.85 | XR2211 | 5.25 | XR4558 | .75 |
| XR1488 | 1.95 | XR2212 | 4.35 | XR4739 | 1.15 |
| XR1489 | 1.95 | XR2240 | 3.45 | XR4741 | 1.47 |

# DIODES

| TYPE | VOLTS | W | PRICE |
|---|---|---|---|
| 1N746 | 3.3 | 400m | 4/1.00 |
| 1N751 | 5.1 | 400m | 4/1.00 |
| 1N752 | 5.6 | 400m | 4/1.00 |
| 1N753 | 6.2 | 400m | 4/1.00 |
| 1N754 | 6.8 | 400m | 4/1.00 |
| 1N757 | 9.0 | 400m | 4/1.00 |
| 1N759 | 12.0 | 400m | 4/1.00 |
| 1N959 | 8.2 | 400m | 4/1.00 |
| 1N965 | 15 | 400m | 4/1.00 |
| 1N5232 | 5.6 | 500m | 4/1.00 |
| 1N5234 | 6.2 | 500m | 4/1.00 |
| 1N5235 | 6.8 | 500m | 4/1.00 |
| 1N5236 | 7.5 | 500m | 4/1.00 |
| 1N5242 | 12 | 500m | 4/1.00 |
| 1N5245 | 15 | 500m | 4/1.00 |
| 1N456 | 25 | 40m | 6/1.00 |
| 1N458 | 150 | 7m | 6/1.00 |
| 1N485A | 180 | 10m | 5/1.00 |
| 1N4001 | 50 PIV 1 AMP | | 12/1.00 |

| TYPE | VOLTS | W | PRICE |
|---|---|---|---|
| 1N4002 | 100 PIV 1 AMP | | 12/1.00 |
| 1N4003 | 200 PIV 1 AMP | | 12/1.00 |
| 1N4004 | 400 PIV 1 AMP | | 12/1.00 |
| 1N4005 | 600 PIV 1 AMP | | 10/1.00 |
| 1N4006 | 800 PIV 1 AMP | | 10/1.00 |
| 1N4007 | 1000 PIV 1 AMP | | 10/1.00 |
| 1N3600 | 50 | 200m | 6/1.00 |
| 1N4148 | 75 | 10m | 15/1.00 |
| 1N4154 | 35 | 10m | 12/1.00 |
| 1N4733 | 5.1 | 1w | 4/1.00 |
| 1N4734 | 5.6 | 1w | 4/1.00 |
| 1N4735 | 6.2 | 1w | 4/1.00 |
| 1N4736 | 6.8 | 1w | 4/1.00 |
| 1N4738 | 8.2 | 1w | 4/1.00 |
| 1N4742 | 12 | 1w | 4/1.00 |
| 1N4744 | 15 | 1w | 4/1.00 |
| 1N1183 | 50 PIV 35 AMP | | 1.60 |
| 1N1184 | 100 PIV 35 AMP | | 1.70 |
| 1N1185 | 150 PIV 35 AMP | | 1.70 |
| 1N1186 | 200 PIV 35 AMP | | 1.80 |
| 1N1188 | 400 PIV 35 AMP | | 3.00 |

# SCR AND FW BRIDGE RECTIFIERS

| | | | | |
|---|---|---|---|---|
| C36D | 15A @ 400V | | SCR(2N1849) | $1.95 |
| C36M | 35A @ 600V | | SCR | 2.25 |
| C106B1 | 3.6A @ 200V | | SCR | .50 |
| 2N2328 | 1.6A @ 300V | | SCR | .50 |
| MDA 980-1 | 12A @ 50V | | FW Bridge Rec. | 1.95 |
| MDA 980-3 | 12A @ 200V | | FW Bridge Rec. | 2.25 |

# TRANSISTORS

| | | | | | |
|---|---|---|---|---|---|
| | | | | 2N3904 | 4/1.00 |
| MPSA05 | 3/1.00 | 2N3055 | .89 | 2N3905 | 4/1.00 |
| MPSA06 | 3/1.00 | MJE3055 | 1.00 | 2N3906 | 4/1.00 |
| TIS97 | 4/1.00 | 2N3392 | 4/1.00 | 2N4013 | .65 |
| TIS98 | 4/1.00 | 2N3398 | 4/1.00 | 2N4123 | 4/1.00 |
| 40409 | 1.75 | PN3567 | 3/1.00 | PN4249 | 4/1.00 |
| 40410 | 1.75 | PN3568 | 4/1.00 | PN4250 | 4/1.00 |
| 40673 | 1.75 | PN3569 | 4/1.00 | 2N4400 | 4/1.00 |
| 2N918 | 2/1.00 | MPS3638A | 4/1.00 | 2N4401 | 4/1.00 |
| 2N2219A | 2/1.00 | MPS3702 | 4/1.00 | 2N4402 | 4/1.00 |
| 2N2221A | 3/1.00 | 2N3704 | 4/1.00 | 2N4403 | 4/1.00 |
| 2N2222A | 3/1.00 | MPS3704 | 4/1.00 | 2N4409 | 4/1.00 |
| PN2222 Plastic | 6/1.00 | 2N3705 | 4/1.00 | 2N5086 | 4/1.00 |
| 2N2369A | 2/1.00 | MPS3705 | 4/1.00 | 2N5087 | 4/1.00 |
| MPS2369 | 4/1.00 | 2N3706 | 4/1.00 | 2N5088 | 4/1.00 |
| 2N2484 | 4/1.00 | MPS3706 | 4/1.00 | 2N5089 | 4/1.00 |
| 2N2906 | 2/1.00 | 2N3707 | 4/1.00 | 2N5129 | 4/1.00 |
| 2N2907 | 3/1.00 | 2N3711 | 4/1.00 | PN5134 | 4/1.00 |
| 2N2925 | 4/1.00 | 2N3724A | .65 | PN5138 | 4/1.00 |
| MJE2955 | 1.25 | 2N3725A | 1.00 | 2N5139 | 4/1.00 |
| 2N3053 | 2/1.00 | 2N3772 | 2.25 | 2N5210 | 4/1.00 |
| | | 2N3823 | 1.00 | 2N5449 | 3/1.00 |
| | | 2N3903 | 4/1.00 | 2N5951 | 2/1.00 |

# CAPACITOR CORNER

**50 VOLT CERAMIC DISC CAPACITORS**

| | 1-9 | 10-99 | 100+ | | 1-9 | 10-99 | 100+ |
|---|---|---|---|---|---|---|---|
| 10 pf | .08 | .06 | .05 | .001µF | .08 | .06 | .05 |
| 22 pf | .08 | .06 | .05 | .0047µF | .08 | .06 | .05 |
| 47 pf | .08 | .06 | .05 | .01µF | .08 | .06 | .05 |
| 100 pf | .08 | .06 | .05 | .022µF | .09 | .07 | .06 |
| 220 pf | .08 | .06 | .05 | .047µF | .09 | .07 | .06 |
| 470 pf | .08 | .06 | .05 | .1µF | .15 | .12 | .10 |

**100 VOLT MYLAR FILM CAPACITORS**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| .001mf | .12 | .10 | .07 | .022mf | .13 | .11 | .08 |
| .0022 | .12 | .10 | .07 | .047mf | .21 | .17 | .13 |
| .0047mf | .12 | .10 | .07 | .1mf | .27 | .23 | .17 |
| .01mf | .12 | .10 | .07 | .22mf | .33 | .27 | .22 |

**+20% DIPPED TANTALUMS (SOLID) CAPACITORS**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| .1/35V | .39 | .34 | .29 | 1.5/35V | .41 | .37 | .29 |
| .15/35V | .39 | .34 | .29 | 2.2/35V | .51 | .45 | .34 |
| .22/35V | .39 | .34 | .29 | 3.3/25V | .53 | .47 | .37 |
| .33/35V | .39 | .34 | .29 | 4.7/25V | .63 | .56 | .45 |
| .47/35V | .39 | .34 | .29 | 6.8/25V | .79 | .69 | .55 |
| .68/35V | .39 | .34 | .29 | 15/25V | 1.39 | 1.25 | .95 |
| 1.0/35V | .39 | .34 | .29 | 22/6V | .79 | .69 | .55 |

**MINIATURE ALUMINUM ELECTROLYTIC CAPACITORS**

| Axial Lead | | | | Radial Lead | | | |
|---|---|---|---|---|---|---|---|
| .47/50V | .16 | .14 | .10 | .47/25V | .15 | .13 | .12 |
| 1.0/50V | .19 | .16 | .12 | .47/50V | .16 | .14 | .13 |
| 3.3/50V | .17 | .15 | .11 | 1.0/16V | .15 | .13 | .12 |
| 4.7/25V | .18 | .15 | .11 | 1.0/25V | .16 | .14 | .13 |
| 10/25V | .18 | .15 | .11 | 1.0/50V | .17 | .15 | .14 |
| 10/50V | .19 | .16 | .12 | 4.7/16V | .15 | .13 | .12 |
| 22/25V | .19 | .16 | .12 | 4.7/25V | .16 | .14 | .13 |
| 22/50V | .24 | .20 | .18 | 4.7/50V | .17 | .15 | .14 |
| 47/25V | .25 | .21 | .19 | 10/16V | .15 | .13 | .12 |
| 47/50V | .29 | .25 | .23 | 10/25V | .16 | .14 | .13 |
| 100/25V | .28 | .24 | .22 | 10/50V | .17 | .15 | .14 |
| 100/50V | .41 | .37 | .34 | 47/50V | .25 | .21 | .19 |
| 220/25V | .39 | .34 | .33 | 100/16V | .21 | .17 | .14 |
| 220/50V | .49 | .45 | .41 | 100/25V | .25 | .23 | .21 |
| 470/25V | .54 | .49 | .45 | 100/50V | .37 | .34 | .31 |
| 1000/16V | .79 | .69 | .61 | 220/16V | .25 | .21 | .19 |
| 2200/16V | .89 | .79 | .69 | 470/25V | .35 | .31 | .27 |

*Start learning and computing for only $129.95 with a Netronics 8085-based computer kit. Then expand it in low-cost steps to a business/development system with 64k or more RAM, 8" floppy disk drives, hard disks and multi-terminal I/O.*

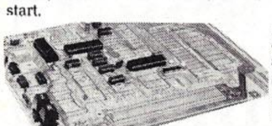
# THE NEW EXPLORER/85 SYSTEM

**Special!** *Full 8" floppy, 64k system for less than the price of a mini! Only* **$1499.95!**

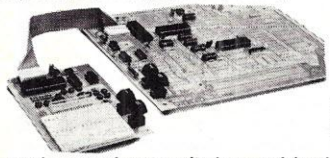
(Also available wired & tested, $1799.95)

Imagine — for only $129.95 you can own the starting level of Explorer/85, a computer that's expandable into full business/development capabilities — a computer that can be your beginner system, an OEM controller, or an IBM-formatted 8" disk small business system. From the first day you own Explorer/85, you begin computing on a significant level, and applying principles discussed in leading computer magazines. Explorer/85 features the advanced Intel 8085 cpu, which is 100% compatible with the older 8080A. It offers onboard S-100 bus expansion, Microsoft BASIC in ROM, plus instant conversion to mass storage disk memory with standard IBM-formatted 8" disks. All for only $129.95, plus the cost of power supply, keyboard/terminal and RF modulator if you don't have them (see our remarkable prices below for these and other accessories). With a Hex Keypad/display front panel, Level "A" can be programmed with no need for a terminal, ideal for a controller, OEM, or a real low-cost start.

*Full 8" disk system for less than the price of a mini (shown with Netronics Explorer/85 computer and new terminal). System features floppy drive from Control Data Corp., world's largest maker of memory storage systems (not a hobby brand!)*



*Level "A" is a complete operating system, perfect for beginners, hobbyists, industrial controller use. $129.95*



*Level "A" With Hex Keypad/Display.*

## LEVEL "A" SPECIFICATIONS

Explorer/85's Level "A" system features the advanced Intel 8085 cpu, an 8355 ROM with 2k deluxe monitor/operating system, and an advanced 8155 RAM I/O . . . all on a single motherboard with room for RAM/ROM/PROM/EPROM and S-100 expansion, plus generous prototyping space.

**PC Board:** Glass epoxy, plated through holes with solder mask. • **I/O:** Provisions for 25-pin (DB25) connector for terminal serial I/O, which can also support a paper tape reader . . . cassette tape recorder input and output . . . cassette tape control output . . . LED output indicator on SOD (serial output) line . . . printer interface (less drivers) . . . total of four 8-bit plus one 6-bit I/O ports. • **Crystal Frequency:** 6.144 MHz. • **Control Switches:** Reset and user (RST 7.5) interrupt . . . additional provisions for RST 5.5, 6.5 and TRAP interrupts onboard. • **Counter/Timer:** Programmable, 14-bit binary. • **System RAM:** 256 bytes located at F800, ideal for smaller systems and for use as an isolated stack area in expanded systems . . . RAM expandable to 64K via S-100 bus or 4k on motherboard.

**System Monitor (Terminal Version):** 2k bytes of deluxe system monitor ROM located at FØØØ, leaving ØØØØ free for user RAM/ROM. Features include tape load with labeling . . . examine/change contents of memory . . . insert data . . . warm start . . . examine and change all registers . . . single step with register display at each break point, a debugging/training feature . . . go to execution address . . . move blocks of memory from one location to another . . . fill blocks of memory with a constant . . . display blocks of memory . . . automatic baud rate selection to 9600 baud . . . variable display line length control (1-255 characters/line) . . . channelized I/O monitor routine with 8-bit parallel output for high-speed printer . . . serial console in and console out channel so that monitor can communicate with I/O ports.

**System Monitor (Hex Keypad/Display Version):** Tape load with labeling . . . tape dump with labeling . . . examine/change contents of memory . . . insert data . . . warm start . . . examine and change all registers . . . single step with register display at each break point . . . go to execution address. Level "A" in this version makes a perfect controller for industrial applications, and is programmed using the Netronics Hex Keypad/Display. It is low cost, perfect for beginners.

## HEX KEYPAD/DISPLAY SPECIFICATIONS

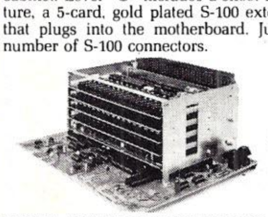
Calculator type keypad with 24 system-defined and 16 user-defined keys. Six digit calculator-type display, that displays full address plus data as well as register and status information.

## LEVEL "B" SPECIFICATIONS

Level "B" provides the S-100 signals plus buffers/drivers to support up to six S-100 bus boards, and includes: address decoding for onboard 4k RAM expansion selectable in 4k blocks . . . address decoding for onboard 8k EPROM expansion selectable in 8k blocks . . . address and data bus drivers for onboard expansion . . . wait state generator (jumper selectable), to allow the use of slower memories . . . two separate 5 volt regulators.

## LEVEL "C" SPECIFICATIONS

Level "C" expands Explorer/85's motherboard with a card cage, allowing you to plug up to six S-100 cards directly into the motherboard. Both cage and card are neatly contained inside Explorer's deluxe steel cabinet. Level "C" includes a sheet metal superstructure, a 5-card, gold plated S-100 extension PC board that plugs into the motherboard. Just add required number of S-100 connectors.



*Explorer/85 With Level "C" Card Cage.*

## LEVEL "D" SPECIFICATIONS

Level "D" provides 4k of RAM, power supply regulation, filtering decoupling components and sockets to expand your Explorer/85 memory to 4k (plus the original 256 bytes located in the 8155A). The static RAM can be located anywhere from ØØØØ to EFFF in 4k blocks.

## LEVEL "E" SPECIFICATIONS

Level "E" adds sockets for 8k of EPROM to use the popular Intel 2716 or the TI 2516. It includes all sockets, power supply regulator, heat sink, filtering and decoupling components. Sockets may also be used for 2k x 8 RAM IC's (allowing for up to 12k of onboard RAM).

## DISK DRIVE SPECIFICATIONS

- 8" CONTROL DATA CORP. professional drive.
- LSI controller.
- Write protect.
- Single or double density.
- Data capacity: 401,016 bytes (SD), 802,032 bytes (DD), unformatted.
- Access time: 25ms (one track).

## DISK CONTROLLER/ I/O BOARD SPECIFICATIONS

- Controls up to four 8" drives.
- 1771A LSI (SD) floppy disk controller.
- Onboard data separator (IBM compatible).
- 2 Serial I/O ports
- Autoboot to disk system when system reset.
- 2716 PROM socket included for use in custom applications.
- Onboard crystal controlled.
- Onboard I/O baud rate generators to 9600 baud.
- Double-sided PC board (glass epoxy.)

## DISK DRIVE CABINET/POWER SUPPLY

- Deluxe steel cabinet with individual power supply for maximum reliability and stability.

## ORDER A COORDINATED EXPLORER/85 APPLICATIONS PAK!

**Beginner's Pak (Save $26.00!)** — Buy Level "A" (Terminal Version) with Monitor Source Listing and AP-1 5-amp Power Supply: (regular price $199.95), now at SPECIAL PRICE: **$169.95** plus post. & insur.

**Experimenter's Pak II (Save $53.40!)** — Buy Level "A" (Hex Keypad/Display Version) with Hex Keypad/Display, Intel 8085 User Manual, Level "A" Hex Monitor Source Listing, and AP-1 5-amp Power Supply: (regular price $279.35), all at SPECIAL PRICE: **$219.95** plus post. & insur.

**Special Microsoft BASIC Pak (Save $103.00!)** — Includes Level "A" (Terminal Version), Level "B", Level "D" (4k RAM), Level "E", 8k Microsoft in ROM, Intel 8085 User Manual, Level "A" Monitor Source Listing, and AP-1 5-amp Power Supply: (regular price $439.70), now yours at SPECIAL PRICE: **$329.95** plus post. & insur.

**ADD A TERMINAL WITH CABINET, GET A FREE RF MODULATOR:** Save over $114 at this SPECIAL PRICE: **$499.95** plus post. & insur.

**Special 8" Disk Edition Explorer/85 (Save over $104!)** — Includes disk-version Level "A", Level "B", two S-100 connectors and brackets, disk controller, 64k RAM, AP-1 5-amp power supply, Explorer/85 deluxe steel cabinet, cabinet fan, 8" SD/DD disk drive from famous CONTROL DATA CORP. (not a hobby brand!), drive cabinet with power supply, and drive cable set-up for two drives. This package includes everything but terminal and printers (see coupon for them). Regular price $1630.30, all yours in kit at SPECIAL PRICE: **$1499.95** plus post. & insur. Wired and tested, only **$1799.95.**

**Special! Complete Business Software Pak (Save $625.00!)** — Includes CP/M 2.0, Microsoft BASIC, General Ledger, Accounts Receivable, Accounts Payable, Payroll Package: (regular price $1325), yours now at SPECIAL PRICE: **$699.95.**

Dept. K10 ***Please send the items checked below:***

- ☐ **Explorer/85 Level "A" kit** (Terminal Version) . . . **$129.95** plus $3 post. & insur.
- ☐ **Explorer/85 Level "A" kit** (Hex Keypad/Display Version) . . . **$129.95** plus $3 post. & insur.
- ☐ **8k Microsoft BASIC** on cassette tape, **$64.95** postpaid.
- ☐ **8k Microsoft BASIC in ROM kit** (requires Levels "B", "D" and "E") . . . **$99.95** plus $2 post. & insur.
- ☐ **Level "B" (S-100) kit** . . . **$49.95** plus $2 post. & insur.
- ☐ **Level "C" (S-100 6-card expander) kit** . . . **$39.95** plus $2 post. & insur.
- ☐ **Level "D" (4k RAM) kit** . . . **$69.95** plus $2 post. & insur.
- ☐ **Level "E" (EPROM/ROM) kit** . . . **$5.95** plus 50¢ p&h.
- ☐ **Deluxe Steel Cabinet** for Explorer/85 . . . **$49.95** plus $3 post. & insur.
  - ☐ Fan For Cabinet . . . **$15.00** plus $1.50 post. & insur.
- ☐ **ASCII Keyboard/Computer Terminal kit:** features a full 128 character set, u&l case; full cursor control; 75 ohm video output; convertible to baudot output; selectable baud rate, RS232-C or 20 ma. I/O, 32 or 64 character by 16 line formats, and can be used with either a CRT monitor or a TV set (if you have an RF modulator) . . . **$149.95** plus $3.00 post. & insur.
  - ☐ **DeLuxe Steel Cabinet** for ASCII keyboard/terminal . . . **$19.95** plus $2.50 post. & insur.
- ☐ **New! Terminal/Monitor:** (See photo) Same features as above, except 12" monitor with keyboard and terminal is in deluxe single cabinet; kit . . . **$399.95** plus $7 post. & insur.
- ☐ **Hazeltine terminals:** Our prices too low to quote — **CALL US**
- ☐ **Lear-Sigler terminals/printers:** Our prices too low to quote: **CALL US**
- ☐ **Hex Keypad/Display kit** . . . **$69.95** plus $2 post. & insur.
- ☐ **AP-1 Power Supply Kit** ±8V @ 5 amps) in deluxe steel cabinet . . . **$39.95** plus $2 post. & insur.
- ☐ **Gold Plated S-100 Bus Connectors** . . . **$4.85** each, postpaid.
- ☐ **RF Modulator kit** (allows you to use your TV set as a monitor) . . . $8.95 **postpaid.**
- ☐ **16k RAM kit** (S-100 board expands to 64k) . . . **$199.95** plus $2 post. & insur.
- ☐ **32k RAM kit** . . . **$299.95** plus $2 post. & insur.
- ☐ **48k RAM kit** . . . **$399.95** plus $2 post. & insur.
- ☐ **64k RAM kit** . . . **$499.95** plus $2 post. & insur.
- ☐ **16k RAM Expansion kit** (to expand any of the above in 16k blocks up to 64k) . . . **$99.95** plus $2 post. & insur. each.
- ☐ **Intel 8085 cpu Users' Manual** . . . **$7.50** postpaid.
- ☐ **12" Video Monitor** (10MHz bandwidth) . . . **$139.95** plus $5 post. & insur.
- ☐ **Beginner's Pak** (see above) **$169.95** plus $4 post. & insur.
- ☐ **Experimenter's Pak** (see above) . . . **$219.95** plus $6 post. & insur.
- ☐ **Special Microsoft BASIC Pak Without Terminal** (see above) . . . **$329.95** plus $7 post. & insur.
- ☐ Same as above, plus **ASCII Keyboard Terminal With Cabinet, Get Free RF Modulator** (see above) . . . **$499.95** plus $10 post. & insur.
- ☐ **Special 8" Disk Edition Explorer/85** (see above) . . . **$1499.95** plus $26 post. & insur.
  - ☐ **Wired & Tested** . . . **$1799.95** plus $26 post. & insur.
- ☐ **Extra 8" CDC Floppy Drives** . . . **$499.95** plus $12 post. & insur.
  - ☐ **Cabinet & Power Supply For Drive** . . . **$69.95** plus $3 post. & insur.
- ☐ **Drive Cable Set-up For Two Drives** . . . **$25** plus $1.50 post. & insur.
- ☐ **Disk Controller Board With I/O Ports** . . . **$199.95** plus $2 post. & insur.
- ☐ **Special: Complete Business Software Pak** (see above) . . . **$699.96** postpaid.
  - SOLD SEPARATELY:
  - ☐ **CP/M 1.4** . . . **$100** postpaid.
  - ☐ **CP/M 2.0** . . . **$150** postpaid.
  - ☐ **Microsoft BASIC** . . . **$325** postpaid.
- ☐ **Intel 8085 cpu User Manual** . . . $7.50 **postpaid.**
- ☐ **Level "A" Monitor Source Listing** . . . **$25** postpaid.

**Continental U.S.A. Credit Card Buyers Outside Connecticut**

**CALL TOLL FREE: 800-243-7428**

To Order From Connecticut Or For Technical Assistance, call (203) 354-9375

Total Enclosed (Conn res. add sales tax) $ ______

Paid By:

☐ Personal Check ☐ Cashier's Check/Money Order

☐ **VISA** ☐ **Master Charge** (Bank No. _ _ _ _)

Acct. No. ______________ Exp. Date ______

Signature ______________

Print

Name ______________

Address ______________

City ______________

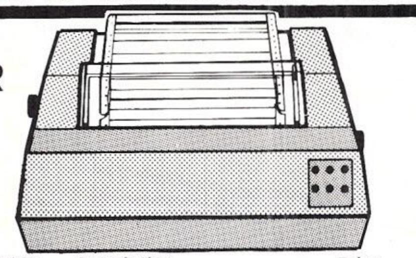
State ______________ Zip ______________

**NETRONICS Research & Development Ltd.**
**333 Litchfield Road, New Milford, CT 06776**

# HOBBYWORLD® ELECTRONICS, INC.

**Call Toll-Free: USA (800) 423-5387 In California: (800) 382-3651**
**Local & Outside USA: (213) 886-9200**

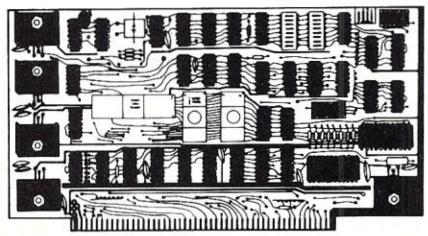
## EMAKO 22 MICROPRINTER



A dependable, low cost addition to your computer system, featuring a 9 x 7 dot matrix character format, bi-directional printing at 125 CPS, and sprocket feed paper mechanism. Line length is selectable at 40, 80, or 132 characters per line. Forms may be loaded from the bottom or rear. Available with parallel or asynchronous serial interfacing. Wt 22 lbs.

| Cat No. | Description | Price |
|---|---|---|
| 2455 | Parallel Interface | $834.75 |
| 2456 | RS-232C Serial Interface | $894.00 |

## SSM CB1-A 8080 CPU Board

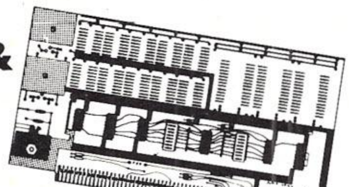


Just add an I/O board and it's a computer! 256 bytes of on board RAM, with option for 2K of on board PROM. Includes a power-on, preset jump circuit, and MWRITE is available, allowing use without a front panel. There's a parallel input port with status, and AIP controlled addressing: or PROM in 2K blocks increments; RAM in 256 byte increments; input port for addresses 0-31 in decimal.

**$115.75 kit**

| | | |
|---|---|---|
| Cat No. 1403 | CB1-A Kit | $115.75 |
| Cat No. 1441 | CB1-A Bareboard | $27.95 |

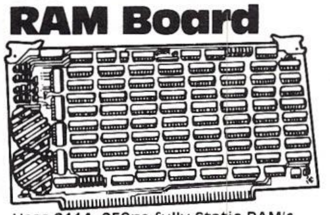
## SSM OB1 Vector Jump & Prototyping Card



Plug compatible for S-100 bus systems, features full 16 bit vector jump address with dip selection of 8080 or Z80. Can be set to jump on Power-on-clear, reset, or both. Prototyping areas on the card for ten 16-pin IC's, three 24-28 pin IC's, and two spare regulator patterns.

| | | |
|---|---|---|
| Cat No. 1429 | OB1 kit | $41.95 |
| Cat No. 1430 | OB1 A&T | $67.95 |
| Cat No. 1431 | OB1 bareboard | $24.95 |

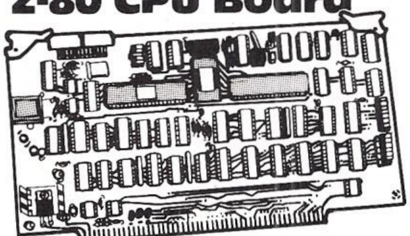
## CCS 32K Static RAM Board



Uses 2114, 250ns fully Static RAM's, Bank selectable in 8K blocks. Enable/Disable on power up or Reset. Compatible with North Star, Alpha Micro, Cromeco, etc. Also front panel compatible, addressable in 8K blocks. Selectable wait state. Wt. 1 lb.

Cat No. 2644 A&T **$649**

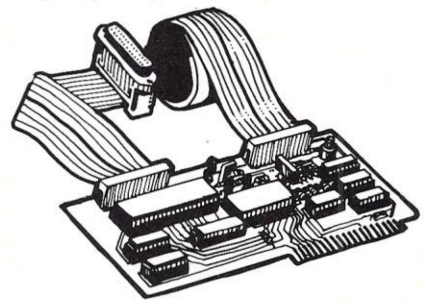
## CCS Z-80 CPU Board



An all new Z-80 CPU board loaded with such great features as S-100/Altair/Imsai compatibility, Power-on jump to any Memory address, selectable Z-80 monitor ROM, selectable MI wait states, full handshake, auto band (2 baud-56K baud) selection, selectable port address, separate baud rate oscillator and on board RS-232 100% disable option serial port. This board also boasts front panel support compatiblity, Z-80 NMI capability, phantom line capability, Z-80 interrupt capability and status valid on Data Lines during psnc. Wt. 3 lbs.

Cat No. 2646 **$299**

## SSM AIO Apple II Serial & Parallel Interface



Allows maximum flexibility for interfacing Apple II with peripherals (printers, plotters, terminals, modems, etc.) Communicates with both serial and parallel devices, and can interface with both at the same time. Complete with software and firmware for serial and parallel communications.

- **Features one RS232 serial interface with:** Three handshaking lines (RTS, CTS, DCD); Nine standard baud rates from 110 to 19,200 baud, including 134.5 baud for selectrics; additional baud selectable through external input; baud rates rotary switch selectable, no jumpers required; serial communication modes software controlled.
- **Two bi-directional 8-bit parallel ports with:** Four additional interrupt and handshaking lines; interface configuration software controlled. Includes on-board firmware for controlling serial interface and software for driving parallel printers. Firmware for parallel interface control optional. Serial and parallel interface cable assemblies included. Includes comprehensive manual and application notes.

| | | |
|---|---|---|
| Cat No. 1918 | AIO Kit | $135.00 |
| Cat No. 1919 | AIO A & T | $175.00 |

## CCS 2422 Disk Controller

This disk controller is equipped with a soft sector format, will support single and double density formats, and supports up to four 5¼" and/or 8" single or double sided drives. It has ROM controlled addressing for I/O mapped and/or (optional) memory mapped operation, fastseek capability for voice-coil type drive, adjustable write precompensation, digital phase-locked data separator, selectable auto-wait on Data or Control port and on-board 2K Byte Boot/program ROM (2716). A copy of CP/M 2.2 is included.

Cat No. 2645 **$399**

## CCS 64K Dynamic RAM Board

Uses low power 4116 Dynamic RAM's, Bank selectable in 16K blocks, bank Enable/Disable on power-up or reset, "fail safe" modes for transparent refresh on 8080 or Z-80, 4mhz operation, phantom line capability and compatible with front panel systems.
Wt. 12 oz.

Cat No. 2647 A&T **$699**

**19511 BUSINESS CTR. DR. DEPT K10 NORTHRIDGE, CA. 91324**

# Power Supplies! Power Supplies! Power Supplies! SOLID STATE!! (5)

# We got 'em! Take your pick . . .

These units are ideal for micro computers. They have been removed from equipment, checked out and guaranteed.

1—5 volts @ 8 amps + 12 volts @ 2 amps + 6 volts @ 75 MA. Power supply has a 3-wire line cord and fused. Dimensions: 10½" × 5½" × 4½". Shipping weight: 16 lbs. . . . . 37.50 ea. 2/70.00

2—Model 818, 5 volts at 15 amps + 12 volts at 4 amps-12 volts at 2 amps. (with line cord). . . . . 35.00 ea. 2/65.00

3— + 5 volts at 5 amps ± 12 volts at 500 ma. + 6 volts at 25 ms. (line cord included). . . . . 32.95 ea. 2/60.00

**4—Elexon, multi output. Input: 120/240 AC, ±10%, 47-63 hz; output: 1) 12V, 1.5A, DC, OVP; 2) 12V, 1.5A, D.C., OVP. New, in box with operating instructions. . . . . 31.50**

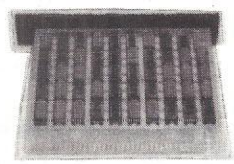
**5—Power Design, Model 1210, constant voltage, DC. P.S. input: 105-125 A.C., 55 to 440 hz. Output: 1-12 volts, 0-10 amps, DC. continuously adjustable output voltage and current limiting. . . . . . 139.00**

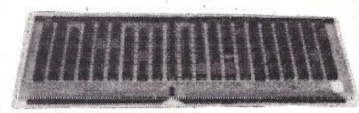
## COMPUTER GRADE CAPACITORS . . .

| | | | | | |
|---|---|---|---|---|---|
| 18,000 mfd 10 VDC | 1.25 | 11,000 mfd 25 VDC | 1.50 | 4,000 mfd 75 VDC | 1.75 |
| 4,400 mfd 20 VDC | 1.00 | 35,000 mfd 35 VDC | 3.50 | 1,000 mfd 100 VDC | 1.00 |
| 46,000 mfd 20 VDC | 2.50 | 10,000 mfd 50 VDC | 2.50 | 6,800 mfd 100 VDC | 3.50 |
| 3.000 mfd 25 VDC | 1.00 | 22,000 mfd 60 VDC | 3.75 | 4,700 mfd 150 VDC | 3.75 |

## WIRE WRAP BOARDS

These boards are pre-wired and removed from equipment. Easy to unwrap for setting up your own board, contains mostly 14-pin IC sockets with individual pin connections. Each board has VCC and ground planes.





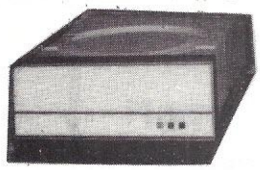
Smaller board measures 6½" × 6" and has 40 to 50 sockets.
Larger board measures 13½" × 6" and has 75 to 100 sockets.

**Reduced prices** **$7.50 ea. 2/$14.00**
**$12.50 ea. 2/$23.00**

## DIABLO System Disc Drive

SERIES 40, MODEL 43

100 tracks per inch, total capacity of 50 megabits, w/Model 429 power supply, sector counter, 24 sectors, 1 fixed disc, 1 removable disc, average access time 38 ms, PPM: 2600, dimensions: 10 5/16" high, fits in standard rack, equipped with full extension slides, excellent used condition. Shipped freight collect.



**$2495**

HEWLETT PACKARD model 200CD/rack mounted AUDIO OSCILLATOR freq:5hz to 600khz output: 160mw $165.00

HEWLETT PACKARD model 400D ANALOG VACUUM TUBE VOLTMETER freq: 10hz to 4mhz voltmeter range: 1mv to 300vac in 12 ranges $85.00

## TRANSFORMERS

**ISOLATION STEP-DOWN TYPE**

Primary: 230/115V, 50/60 CPS, Secondary: 115 volts output 250 VA.

**$13.95** EACH

## IMC MAGNETICS SUPER BOXER FANS

Unused, Model WS2107FL—310, 220/240 VAC, .3 amps, 50/60 hz, 4 11/16" × 4 11/16" × 1 1/2"

**$8.95**

**Clock Crystal Oscillators—TTL, Vectron, type CO-231T. Crystal freq. 4.9152 mhz. Input voltage 5 VDC ±. Output: Drives 10 TTL Loads Logic "0": 0.4V max., sink 16ma. Logic "1" 2.4V min source 2 ma. (above 50 mhz drives 2 Schottky TTL loads). Tuning adjust. with nominal range of ±30 ppm below 25 mhz and 15 ppm above 25 mhz. R.F.E. 1½" × 1½" × ½". . . . . $13.95**

## SG-132 SWEEP SIGNAL GENE OR

FREQ: 15 TO 400 M

Output: AM & FM: CW. F ... at any frequency. Crysta ... , 5mhz or ± 10B. Frequency ac ... oscilloscope for observing waveforms.

**SOLD OUT**

**$329**

## TRENDLINE PHONES

Manufactured by I.T.T.

These units have rotary dials. Colors are: white, black, red, and green. They are packaged and have 6-foot cord and installation instructions. Used, but in good operating condition.

**34.50 WALL TYPE**

Minimum order $25.00. Items offered subject to prior sale. FOB, Brockton, Mass. Money order or check w/order. Shipments and handling add 5%. Shipments by parcel post or UPS. No CODs. Mass. residents add 5% sales tax.

# WALLEN

ELECTRONICS CO. INC. Tel: (617) 588-6440-6441
45 108 SAWTELL AVE., BROCKTON, MA. 02402

ELECTRONIC COMPONENTS
TEST EQUIPMENT
CONNECTORS—WIRE

# New Releases for the TRS-80*

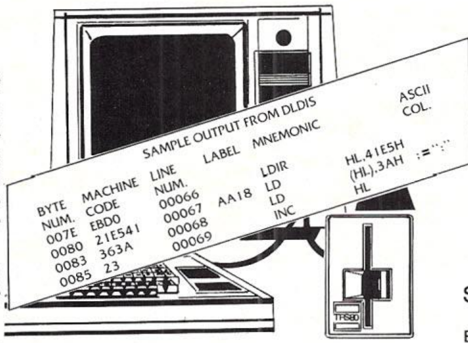
## Utilities

We're proud to present three disassemblers for the TRS-80. For speed and simplicity, we recommend The Disassembler. For complex disassemblies, especially if you wish to make alterations, you may prefer one of our Labeling Disassemblers, either TLDIS or DLDIS.

### TLDIS & DLDIS

You've bought a super machine-code program, but now wonder how it works. Maybe you even used a quick PEEK routine to glance through it when it was in memory. If so, you definitely noticed the complete lack of comments in the code, making it almost impossible for you to decipher and understand it.

Well, Instant Software's Labeling Disassemblers are the answer to your problem.




TLDIS (Tape-based Labeling DISassembler) and DLDIS (Disk-based Labeling DISassembler) are three-pass, label-assigning disassemblers which assign labels (where appropriate) to the routines in a machine-language program. Their output is almost identical to that of a hand-assembled source code.

You can send the disassembly to a lineprinter (Radio Shack parallel port) for either TLDIS or DLDIS. (The difference between these utilities is the storage mode of the disassembly.)

TLDIS can send the disassembly to cassette tape, DLDIS can send it to disk; both send it to the video monitor. The stored disassembly from TLDIS may be reassembled with Radio Shack's EDTASM(TM)—the disassembly from DLDIS, with Apparat's extension of EDTASM(TM).

Because of the use of labels, it is a simple matter to change any object code program by disassembling it and then making changes to the resultant source code, without losing track of jump/load addresses. Labels start with "AA00" and increment up, in even numbered steps (AA02, AA04, etc.). The odd numbers (AA01, AA03, etc.) are left for you to use for the source code during reassembly.

The printing of the disassembly may be temporarily halted by using [SHIFT] @ (just as in BASIC) or it may be ended by pressing the [BREAK] key. It also has a comments column to display ASCII charachters used in a LD or CP opcode.

Because TLDIS and DLDIS work only on in-memory programs, they may be relocated in memory to avoid conflict with the program you disassemble.

The next time you need to "climb inside" a machine-code program, take DLDIS or TLDIS with you. We promise that it will be an easier journey.

**Order No. 0230R (TLDIS) $14.95**
**Order No. 0231RD (DLDIS) $19.95**

### The Disassembler

This is a single-pass, hex-notation disassembler that will send its output either to tape or to a lineprinter (Radio Shack parallel port). The tape output is directly compatible with Tandy's EDTASM(TM). Thus, you can take an object code tape, disassemble and output it to tape, then use EDTASM(TM) to add, delete, change and even re-assemble your new version.

In addition, it displays the *displacement* and *absolute* address of any relative jumps made by the disassembled program. It also displays any ASCII characters used in a LD or CP opcode.

Sample output from the Disassembler

| BYTE NUM. | MACHINE CODE | LINE NUM. | MNEMONIC | COMMENTS COLUMN | |
|---|---|---|---|---|---|
| 706E | 22057B | 00053 | LD | (7B05H),HL | |
| 7071 | 183B | 00054 | JR | $+3DH | ;70AEH |
| 7073 | FE52 | 00055 | CP | 52H | ;="R" |
| 7075 | 2007 | 00056 | JR | NZ,$+09H | ;707EH |
| 7077 | CD8F70 | 00057 | CALL | 708FH | |

H means the number is HEX
$ means current location counter.

Since the Disassembler works only on in-memory programs, it has been made relocatable so that you may move it around in memory to avoid conflict with the program you wish to disassemble. As an added option, you may also jump to memory locations and transfer control between Disassembler and other utility programs in your computer.

The Disassembler, use it to examine and analyze *any* machine-code program!

**Order No. 0232R $9.95**

### Terminal-80

The Terminal-80 package lets your TRS-80 communicate with the rest of the world. These programs give you control of the RS-232 port of your Expansion Interface.

You can connect one or more serial terminals to your TRS-80. Your computer will accept input from the RS-232 port just as if it were entered from the keyboard. Thus, you can use your computer from a remote terminal without having to move your equipment.

The TRS-80 can also be transformed into a "dumb" terminal. You can use it in a time-sharing situation to talk to "big" computers via a modem. All data that you type in will go out through the RS-232 port and all incoming data will be displayed on the screen.

You can transfer programs over the phone lines. Just load a program into the TRS-80. The LPRINT/LLIST command will transfer the program to a receiving computer via the RS-232 port.

Using the upper/lowercase modification of the TRS-80 is simplified. (You must have the modification kit installed first or follow the detailed instructions included in this package.) Control characters in Level II and Disk BASIC will be properly displayed and all functions such as CHR$ will work correctly.

This package even includes a BASIC program to set the baud rate. You won't have to tear apart your Expansion Interface if you use more than one configuration.

There are thousands of TRS-80 computers in the world. Let's get together and talk to each other—with the Terminal-80 from Instant Software.

This package requires the following minimum system:

1. A TRS-80 with 16K of memory.
2. An Expansion Interface.
3. An RS-232 Serial Interface (e.g., Radio Shack's No. 26-1145 or the equivalent).
4. An optional upper/lowercase modification kit.

**Order No.0130R (cassette-based) $24.95.**

**WRITE FOR OUR NEW CATALOG!**

# Instant Software™

40

*A trademark of Tandy Corporation

PETERBOROUGH, N.H. 03458
603-924-7296

# WAMECO

## THE COMPLETE PC BOARD HOUSE
## EVERYTHING FOR THE S-100 BUSS

* **FPB-1** FRONT PANEL BOARD FOR 8080A AND Z80 SYSTEMS IMSAI COMPATIBLE.
  PCBD ........ $56.95 KIT ........ $175.00
* **MEM-2** 16K RAM 2114's. ADDRESSABLE IN 4K BOUNDARIES.
  PCBD .... $33.95 KIT (LESS RAMS) .... $80.95
* **EPM-2** 16/32K ROM USES 2716 OR 2708. ADDRESSABLE IN 4K BOUNDARIES.
  PCBD .... $33.95 KIT (LESS ROMS) .... $74.95
* **CPU-1** 8080A PROCESSOR BOARD WITH VECTOR INTERRUPT.
  PCBD ........ $33.95 KIT ........ $124.95
* **IOB-1** I/O BOARD. ONE SERIAL, TWO PARALLEL WITH CASSETTE. PCBD ....... $33.95
* **FDC-1** FLOPPY DISC CONTROLLER BOARD USES 1771. PCBD ....... $45.95
* **QMB-12** 13 SLOT MOTHER BOARD.
  PCBD ........ $42.95 KIT ........ $125.95
* **QMB-9** 9 SLOT MOTHER BOARD.
  PCBD ........ $35.95 KIT ....... $109.95
* **PTB-1** POWER SUPPLY AND TERMINATOR BOARD.
  PCBD ........ $29.95 KIT ........ $49.95
* **RTC-1** REAL TIME CLOCK BOARD WITH TWO INTERRUPTS.
  PCBD ........ $29.95 KIT ........ $79.95
* **MEM-1** 8K RAM, USES 2102's.
  PCBD .... $33.95 KIT (LESS RAM) .... $71.95
* **EPM-1** 4K 170Z BOARD.
  PCBD .... $29.95 KIT (LESS ROM) .... $59.95

FUTURE PRODUCTS: 80 CHARACTER VIDEO BOARD.
Z-80 CPU BOARD WITH ROM, 8 PARALLEL PORT I/O BOARD

**DEALER INQUIRIES INVITED, UNIVERSITY DISCOUNTS AVAILABLE**
**AT YOUR LOCAL DEALER**

MOST PRODUCTS FOR IMMEDIATE SHIPMENT. NO 4-8 WEEK DELAYS REQUIRED FOR OTHERS.

WMC inc.

WAMECO, INC., P. O. BOX 877 • 455 PLAZA ALHAMBRA • EL GRANADA, CA 94018 • (415) 726-6378

---

**CALIFORNIA COMPUTER SYSTEMS**

**16K** RAM BOARD. Fully buffered addressable in 4K blocks. IEEE standard for bank-addressing 2114's. PCBD ................$28.95 Kit 450 NSEC ....$249.95

**PT-1** PROTO BOARD. Over 2,600 holes 4" regulators. All S-100 buss functions labeled, gold fingers. PCBD ........ $28.95

**PT-2** PROTO BOARD. Similar to PT-1 except set-up to handle solder tail sockets. PCBD ..$28.95

**CCS MAIN FRAME.** Kit (S-100) .................. $339.95

**APPLE EXTENDER.** Kit ........ $22.95

APPLE IEEE INSTRUMENTATION INTERFACE KIT 7490. Kit ........ $275.00

ARITHMETIC PROCESSOR FOR APPLE 7811A. Kit ........ $350.00

APPLE ASYNCHRONOUS SERIAL INTERFACE 7710A. Kit ........ $89.95

APPLE SYNCHRONOUS SERIAL INTERFACE 7712A. Kit ........ $89.95

ALL OTHER CCS PRODUCTS AVAILABLE

**SSM**

**PB-1** 2708 & 2716 Programming Board with provisions for 4K or 8K EPROM. No external supplies required. Textool sockets. Kit ........ $143.00

**CB-1A** 8080 Processor Board. 2K of PROM 256 BYTE RAM power on/rest Vector Jump Parallel port with status. Kit ....$146.00 PCBD ...... $31.95

**VB-3** 80x24 VIDEO BOARD. Graphics included. 4MHZ ........ $379.95

**IO-4** Two serial I/O ports with full handshaking 20/60 ma current loop: Two parallel I/O ports. Kit ........ $168.00 PCBD ........ $31.95

**VB-IC** 64 x 16 video board, upper lower case Greek composite and parallel video with software, S-100. Kit ........ $143.00

**CB-2** Z80 CPU BOARD. Kit ........ $199.95

**AIO** APPLE SERIAL/PARALLEL ........ $144.95

ALL OTHER SSM PRODUCTS AVAILABLE

---

**WMC inc. WAMECO INC.**

**FDC-1** FLOPPY CONTROLLER BOARD will drive shugart, pertek, remic 5" & 8" drives up to 8 drives, on board PROM with power boot up, will operate with CPM™ (not included). PCBD ........ $43.95

**FPB-1** Front Panel. IMSAI size, hex displays. Byte, or instruction single step. PCBD ........ $48.50

**MEM-1A** 8K x 8 fully buffered, S-100, uses 2102 type rams. PCBD ........ $28.95

**QM-12** MOTHER BOARD, 13 slot, terminated, S-100 board only ........ $39.95

**CPU-1** 8080A Processor board S-100 with 8 level vector interrupt. PCBD ........ $28.95

**RTC-1** Realtime clock board. Two independent interrupts. Software programmable. PCBD....... $25.95

**EPM-1** 1702A 4K Eprom card. PCBD ........ $25.95

**EPM-2** 2708/2716 16K/32K EPROM CARD. PCBD ........ $28.95

**QM-9** MOTHER BOARD. Short Version of QM-12. 9 Slots. PCBD ........ $33.95

**MEM-2** 16K x 8 Fully Buffered 2114 Board. PCBD ........ $28.95

**PTB-1** POWER SUPPLY AND TERMINATOR BOARD. PCBD ........ $28.95

**IOB-1** SERIAL AND PARALLEL INTERFACE. 2 parallel, one serial and cassette. PCBD ........ $28.95

| | | | |
|---|---|---|---|
| 2708 | $ 9.49 | 2114L 450 NSEC | $5.99 |
| 2716 | $35.95 | 2114L 200 NSEC | $6.99 |

# MIKOS

(415) 726-7593
**P. O. Box 955 • El Granada, CA 94018**
Please send for IC, Xistor and Computer parts list

---

**OCT. SPECIAL SALE ON PREPAID ORDERS**
(Charge cards not included on this offer)

**RAM SALE**
MEM-2, PCBD ........ $25.95
MEM-2 + PARTS LESS RAM ....... $49.95
MEM-2 MIKOS #7 450 NSEC..... $199.95

**MIKOS PARTS ASSORTMENT WITH WAMECO AND CYBERCOM PCBDS**

**MEM-2** with MIKOS #7 16K ram with L2114 450 NSEC ........ $229.95

**MEM-2** with MIKOS #13 16K ram with L2114 200 NSEC ........ $249.95

**CPU-1** with MIKOS #2 8080A CPU ........ $99.95

**QM-12** with MIKOS #4 13 slot mother board ........ $110.95

**RTC-1** with MIKOS #5 real time clock........ $65.95

**EMP-1** with MIKOS #10 4K 1702 less EPROMS ........ $ 49.95

**EPM-2** with MIKOS #11 16-32K EPROMS less EPROMS ........ $65.95

**QM-9** with MIKOS #12 9 slot mother board ........ $99.95

**FPB-1** with MIKOS #14 all parts for front panel ........ $144.95

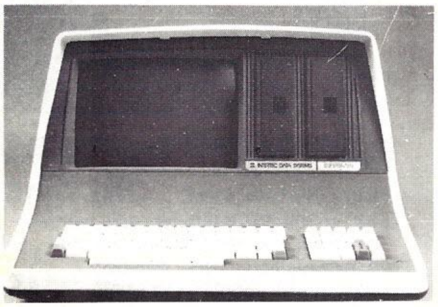
MIKOS PARTS ASSORTMENTS ARE ALL FACTORY MARKED PARTS. KITS INCLUDE ALL PARTS LISTED AS REQUIRED FOR THE COMPLETE KIT LESS PARTS LISTED. ALL SOCKETS INCLUDED.

**LARGE SELECTION OF LS TTL AVAILABLE**
PURCHASE $50.00 WORTH OF LS TTL AND GET 10% CREDIT TOWARD ADDITIONAL PURCHASES.
*PREPAID ORDERS ONLY.*

VISA or MASTERCHARGE. Send account number, interbank number, expiration date and sign your order. Approx. postage will be added. Check or money order will be sent post paid in U.S. If you are not a regular customer, please use charge, cashier's check or postal money order. Otherwise there will be a two-week delay for checks to clear. Calif. residents add 6% tax. Money back 30-day guarantee. We cannot accept returned IC's that have been soldered to. Prices subject to change without notice. **$10 minimum order. $1.50 service charge on orders less than $10.00.**

# Computers, Disk Systems

## SUPERBRAIN®
### By INTERTEC



32K or 64K (Double or Quad Density units available). Uses two Z-80 CPU's. Commercial-type terminal with 12" monitor. Dual double density minifloppies. Over 350 kilobytes of storage (twice that with quad density drives). Two serial RS232 ports, I/O ports standard. Expandable with optional S-100 S-100 interface. Comes with CP/M™ 2.2 operating system. MiniMicroMart includes BASIC interpreter and can supply a wide range of CP/M Development and Application software.

w/32K Double Density, List $2995 . **$2685**
w/64K Double Density, List $3345 . . . . . **$2883**
w/64K Quad Density, List $3995 . . . . . . **$3595**
W/64K Quad — MiniMicroMart Upgrade Special. . . . . . . . . . . . . . . . . . . . . **$3395**

## MICROMATION

A 64K complete computer with dual density 8" floppies (1 megabyte). Rack or vertical mounting. Systems with double-sided drives, hard disks, and multi-user (MP/M).

Z+ 100 64K RAM,Computer, $2495. . **$2099**
Z+120 Includes two 8" disks, $4995 . . . **$4199**

"Z" system features new distributed processing multi-user concept with one Z-80 per user, with Z-80 for MP/M (Master Satellite concept).

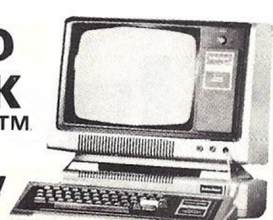
**AS LOW AS $11,899!**

## SD SYSTEMS

SDS-100, w/32K RAM, $6995 . . . . . **$5945**
SDS-200, List $8995 . . . . . . . . . . . . . . . . **$7645**

## RADIO SHACK TRS-80™

***10% OFF!***



## INTERSYSTEMS
formerly ITHACA AUDIO

DPS-1, List $1795

**LIMITED TIME $1299***

The new Series II CPU Board features a 4 MHz Z-80A CPU and a full-feature front panel. 20-slot actively terminated motherboard, with 25 amp power supply (50/60 Hz operation, incl. 68 cfm fan).

**COMPLETE SYSTEM** with InterSystem 64K RAM, I/O Board w/priority interrupt and double density disk controller board. Full 1-year warranty, List $3595

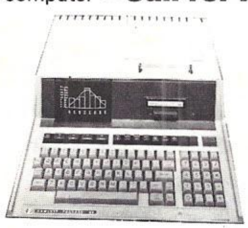
**LIMITED TIME $2895***

Above without disk controller,
List $3195 . . . . . . . . . . **LIMITED TIME $2539***

** Prices good until September 15, 1980.*

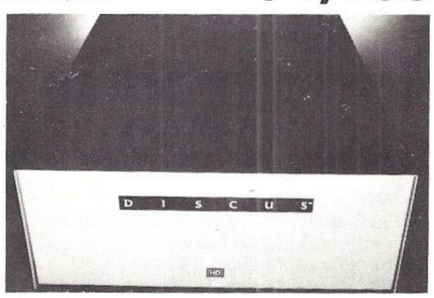
## HEWLETT-PACKARD
## HP-85A

Desk-top computer — ***Call for Price!***



## MORROW
### THINKER TOYS®
### DISCUS M26™

26 megabytes of formatted storage
List $4,995 **$4,199**




## THINKER TOYS®
## DISK SYSTEMS

Now includes CP/M® 2.2

Discus 2D, List $1199. . . . . . **$1019**

Discus 2D, dual-drive, List $1994 . . . . . . **$1694**
Discus 2+2, Assem., List $1549. . . . . . . **$1319**
Dual Discus 2+2, Assem., $2748. . . . . . **$2335**

*All Morrow systems now include CP/M® 2.2*

✓226

## NORTH STAR
## DOUBLE DENSITY
## CONTROLLER BOARDS

Kit, List $399
**OUR PRICE $329**
Assembled and Tested, List $499. . . . . . . . **$399**

*In Stock — First Time in 2 Years!*

## FANTASTIC SAVINGS on a "QUAD" DENSITY HORIZON UPGRADE

North Star Double Density Controller Board (see above) and a quad density MPI-52 (features superior disk handling and door mechanism.

MDS-H-MQ/K Kit form
List $999 **OUR PRICE $699**

MDS-H-MQ/A Assembled form, List $1099 **$759**

Shipping and insurance: Add $6.

## NORTH STAR MDS-A
## Double Density Mini Floppy Disk System

**Double Density**, Kit
List $799 **OUR PRICE $669**
Assembled and Tested. . . . . . . . **$719**
**Quad Version**, Kit, List. . . . . . . **$836**
Assembled, List $1099. . . . . . . . **$896**

*Above MDS-A units do not include cabinet or power supply.*
Shipping and Insurance: Add $7.50.

## *Super Special!*

**North Star Controller Board, Drive, Cabinet, and Power Supply $709**

Complete system similar to above but also includes a cabinet and an assembled/tested power supply for the drive (silver finish). Your choice of Shugart SA-400 or MPI-51 Double Density Drive or MPI-52 quad density drive (MPI drives feature improved door and disk handling mechanism).

w/Controller Bd. kit, SA-400. . . . . . . . . . . **$709**
w/Controller Bd. kit, MPI-51. . . . . . . . . . . **$709**
w/Controller Bd. kit, MPI-52. . . . . . . . . . . **$809**
w/Assembled Bd. and SA-400. . . . . . . . . . **$769**
w/Assembled Bd. and MPI-51. . . . . . . . . . **$769**
w/Assembled Bd. and MPI-52. . . . . . . . . . **$869**

Shipping and Insurance: Add $6.

For converting existing Horizon 2 to quad, order additional MPI-52

MPI-52 Quad Density Drive . . . . . . . . . . . . **$379**

*Turn Page*

# Terminals and Printers!

## TELEVIDEO TVI-912C

Upper and lower case, 15 baud rates: 75 to 19,000 baud, dual intensity, 24 x 80 character display, 12 x 10 resolution. Numeric pad. Programmable reversible video, auxiliary port, self-test mode, protect mode, block mode, tabbing, addressable cursor. Microprocessor controlled, programmable underline, line and character insert/delete. "C" version features typewriter-style keyboard. List $950

**OUR PRICE $789**

**920C** (with 11 function keys, 6 edit keys and 2 transmission mode keys, List $1030

**ONLY $849**

## Intertec EMULATOR

Software compatible with a Soroc IQ-120, Hazeltine 1500, ADM-3A or DEC VT-52. Features block mode transmission and printer port; 12" anti-glare screen; 18-key numeric keypad; full cursor control. List $895

**OUR PRICE $729**

## Intertec INTERTUBE II

List $995 **ONLY $799**

12" display, 24 x 80 format, 18-key numeric keypad, 128 upper/lower case ASCII characters. Reverse video, blinking, complete cursor addressing and control. Special user-defined control function keys, protected and unprotected fields. Line insert/delete and character insert/delete editing, eleven special line drawing symbols.

## SOROC

**IQ-120**

List $995

**SPECIAL**

**$729**

IQ-140 List $1495

**SPECIAL $1149**

## HAZELTINE

**1500**

**ONLY**

**$879**

| | |
|---|---|
| 1410 w/numeric keypad, List $900 | $749 |
| 1420 w/lower case and numeric pad | 849 |
| 1510, List $1395 | 1089 |
| 1520, List $1650 | 1389 |

## BANTAM 550

**From Perkin-Elmer**

**ONLY**

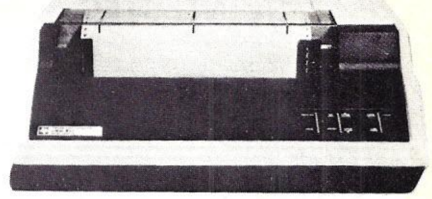
**$799**

with anti-glare CRT **ONLY $829**

## CENTRONICS PRINTERS

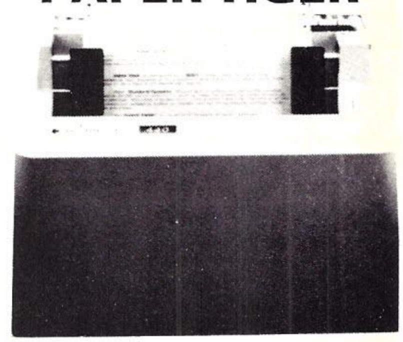
| | |
|---|---|
| **NEW 730**, parallel, friction, tractor | **$679** |
| **NEW 737** parallel, friction, tractor | **$849** |
| 779-2 w/tractor (same as TRS-80 Line Printer I), List $1350 | 1049 |
| 702 120 cps, bi-direct., tractor, VFU | 1995 |
| 703 185 cps, bi-direct., tractor, VFU | 2395 |
| 704 RS232 serial version of 703, $2350 | $1995 |

## TI-810



| | |
|---|---|
| TI-810 Basic Unit, $1895 | **ONLY $1695** |
| TI-810 w/full ASCII (Lower case), vertical forms control, and compressed print | **$1895** |
| TI-745 Complete printing terminal with acoustic coupler, List $1695 | **$1399** |

## PAPER TIGER®



| | |
|---|---|
| **IDS-440 Paper Tiger**, List $995 | **$895** |
| w/graphics option, incl. buffer, $1194 | **$989** |
| TRS-80 cable | **45** |

## NEC SPINWRITER™

Terminal/Keyboard as well as RO Printer Only models available.

***CALL FOR PRICES!***

| | |
|---|---|
| **OKIDATA Microline 80** | **ONLY $649** |
| Tractor Feed Option | **$99** |
| Serial interface | **$89** |
| **AXIOM IMP I** | **$699** |
| **COMPRINT 912** w/parallel interf. | **$559** |
| 912 w/serial interface, List $699 | **$589** |
| **MICROTEK**, List $750 | **$675** |
| **ANADEX** 80-Col. Dot Matrix | **$849** |

Above prices reflect a 2% cash discount (order prepaid prior to shipment). Add 2% to prices for credit card orders, C.O.D.'s, etc. Prices are f.o.b. shipping point. Prices are subject to change and offers subject to withdrawal without notice. **WRITE FOR FREE CATALOG.**

✓50

# MiniMicroMart, Inc.

VISA

master charge

**1618 James Street, Syracuse NY 13203 (315) 422-4467 TWX 710-541-0431**

# MICROCOMPUTING T.M. LIST OF ADVERTISERS



***This advertiser prefers to be contacted directly.**

**For further information from our advertisers, please use the Reader Service card located on the last page.**

*Call our new toll free number for further information.*

*1-800-255-6638*

# The Businessman's Business System

MSI Business Computer Systems offer flexibility and expandability unmatched by any other microcomputer system, large or small. Our SDOS operating system is totally device independent and supports up to four users. This means that you can start with a single user, dual drive, floppy disk system today, and add up to 80 megabytes of hard disk with additional workstations tomorrow. As your business grows, your MSI system grows with you—and your software won't become obsolete.

Perform text processing tasks at one workstation while entering sales orders on another. Add a third workstation in inventory control and a fourth in accounting. That's expandability!!!

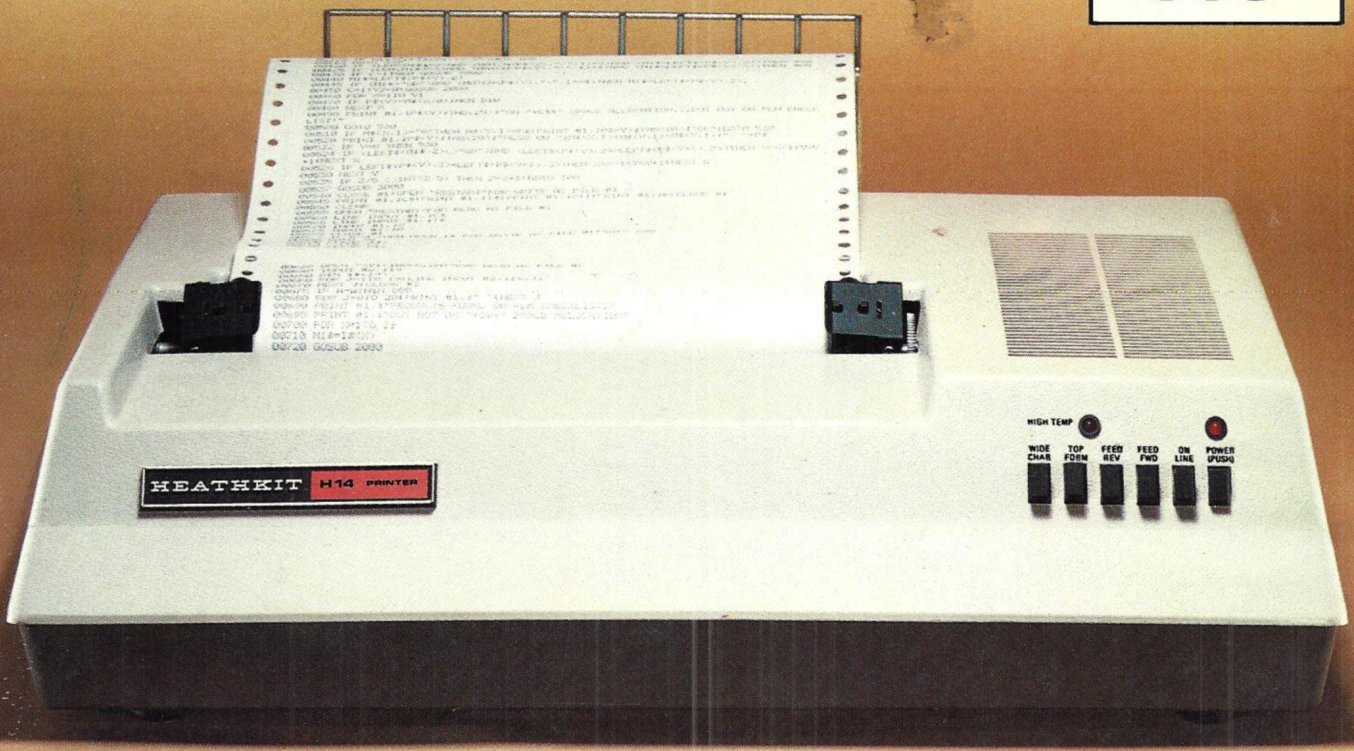
- MSI Inventory Software, with complete Bills of Material, provides a complete inventory control and management system for manufacturers.
- Complete manufacturing forecasting, with production pick lists, allows automatic adjustment of component inventory levels.
- All transactions resulting in any change to the inventory data base are written to audit trail files listing date, time, operator's name, inventory item, and the changes which were made.
- Sales Order Entry/Accounts Receivable Software displays customer balances and credit standing as new orders are entered. Correct product prices and descriptions are obtained from inventory files if desired.
- Invoices are generated automatically as orders are shipped. Customer statements, with aged accounts receivable, are printed on demand.
- Purchase Order Entry/Accounts Payable Software optionally link to inventory program, in order to easily visualize inventory items which are on order.
- General Ledger programs link to the accounts receivable and accounts payable modules for easy updates and posting.
- If your business is expanding and you would like to know how an MSI Computer System can help you make it more profitable, call or write Midwest Scientific Instruments, 220 W. Cedar, Olathe, Kansas 66061, (913) 764-3273, TWX 910 749 6403 (MSI OLAT), TELEX 42525 (MSI A OLAT).

**Small Computers For Big Jobs**

**MSI Midwest Scientific Instruments**

# Very sharp printout at very low payout

**$595***




## The Heath H-14 Printer gives you high-performance features at one of the lowest prices anywhere...

- 5 x 7 dot matrix and high quality impact printhead give you clear, easy-to-read images
- Standard 96-character ASCII set, UPPER and lower case
- Operator or software-selectable line widths; 132, 96, or 80 characters
- Compatibility with any computer having RS-232C or 20 mA current loop serial interface, with handshaking
- Sprocket paper feed, with adjustable spacing, to keep paper moving smoothly
- Sustainable print speed approximately 30 characters per second
- "Paper jammed" and "paper out" signals to prevent loss of data
- Selectable baud rates from 110 to 4800
- Convenience of standard fan-fold paper, 2.5 to 9.5 inches wide
- Chrome wire rack to keep paper neat

The H-14's remarkable price includes connecting cables, paper rack, paper, and ribbon – so you're all ready to run. And service for the H-14 is available at 61 Heathkit Electronic Centers in the U.S. or Canada.

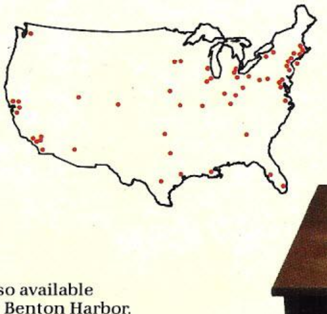
Check out the microprocessor-based H-14 Printer today, in kit form or factory assembled. You'll find complete details in the newest, FREE Heathkit Catalog. Send for yours today, or pick one up at the nearest Heathkit Electronic Center. **Dealer inquiries on assembled units are invited, too.**

**Complete service, so you're never left out in the cold**

**Heath®**
**Computer Systems**

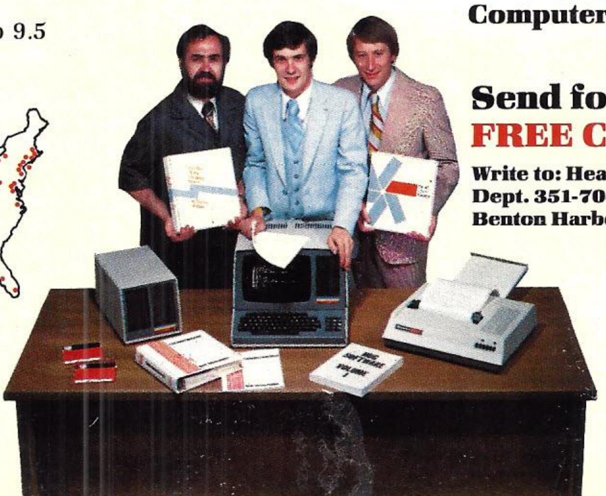
**Visit your Heathkit Service Center**

**H-14 Printers are on display at the 61 Heathkit Electronic Centers throughout the U.S. and Canada. See your telephone white pages for the location nearest you.**



In the U.S. Heathkit Electronic Centers are units of Veritechnology Electronics Corporation.

*In kit form, FOB Benton Harbor, MI. Also available completely assembled at $895.00, FOB Benton Harbor. Prices subject to change without notice.



**Send for** ✓236
**FREE Catalog**

**Write to: Heath Company, Dept. 351-704. Benton Harbor, MI 49022**

CP-191